Universitat de Lleida

TRABAJO FINAL DE GRADO

Estudiante: Mihaela Claudia Diosan

Titulación: Grado en Ingeniería Informática

Título del Trabajo Final de Grado: **Aplicación móvil para ayudar a personas sin recursos**

Director/a: Montserrat Sendín Veloso

Presentación

Mes: Octubre

Año: 2020

Agradecimientos

Quiero dar las gracias a todas las personas que me han ayudado en la realización de este proyecto y en especial a mi tutora, Montserrat Sendín, que me ha guiado durante todo el proceso.

También quiero agradecer la ayuda recibida por parte de los Laboratorios TIC y el Lleida Liquid Galaxy LAB, situados en el Parque Científico de Lleida y su coordinador, Andreu Ibañez, por ofrecerme su ayuda y la de los estudiantes del laboratorio. También darle las gracias por prestarme el equipamiento necesario para realizar las pruebas a lo largo del proyecto.

Por último, quiero agradecer los consejos sobre aspectos legales y de privacidad recibidos por parte del señor Ramon Arnó.

Resumen

El siguiente trabajo de final de grado, consiste en el desarrollo de una aplicación móvil para la plataforma Android que pretende poner en contacto personas sin recursos con personas que están dispuestas a ayudar. Todo esto será posible a través de voluntarios que serán los encargados de buscar y crear nuevos perfiles para las personas que necesitan ayuda. La aplicación, llamada HAPIS (Homeless Aid Panoramic Interactive System) cuenta con una tecnología extra a través de la cual es posible visualizar toda la información en un sistema de pantallas panorámicas único, el Liquid Galaxy.

Cabe destacar que parte del presente proyecto también se ha desarrollado durante la participación en el programa Google Summer Of Code 2020 [1] que será descrito en el siguiente capítulo.

El presente informe se ha redactado siguiendo las normas IEEE [2] para facilitar la comprensión de su lectura.

ÍNDICE DE CONTENIDO

ÍNDICE DE GRÁFICOS

ÍNDICE DE TABLAS

ÍNDICE DE IMÁGENES

1. Introducción

1.1. Contexto

Actualmente, resulta inconcebible un día a día sin teléfono móvil, y más en concreto, sin Smartphone[1]. Debido al auge de estos dispositivos, cada vez más personas utilizan los múltiples servicios que ofrecen. Android es el sistema operativo para dispositivos móviles que ha experimentado mayor crecimiento en los últimos años. En cuanto al desarrollo de las aplicaciones para Android, encontramos multitud de herramientas que mejoran nuestra productividad, nos ayudan a estar en forma, comunicarnos etc., pero hay muy pocas aplicaciones a través de las cuales podemos ayudar de forma rápida y sencilla a los más necesitados y menos aún aplicaciones que permitan la conexión entre las dos partes: las personas que pueden y están dispuestas a ayudar y las personas que necesitan nuestra ayuda.

1.2. Motivación

Tal y como se ha explicado en el contexto, hoy en día existen muy pocas aplicaciones que permiten ayudar a personas que no disponen de cosas tan básicas como comida o ropa. Y menos aún aplicaciones que permiten la fácil comunicación de estos dos tipos de personas.

Con este proyecto se pretende proporcionar una herramienta a través de la cual sea posible todo lo mencionado anteriormente de una manera fácil y sencilla utilizando una aplicación móvil. La aplicación se va a desarrollar para el sistema operativo Android[2] y contará con dos tipos de usuarios: los donantes, que podrán ofrecer ayuda a los más necesitados y los voluntarios, que serán los encargados de buscar y crear nuevos perfiles para las personas necesitadas.

Además, la aplicación contará con una parte extra a través de la cual será posible visualizar la información en un sistema de pantallas panorámicas, Liquid Galaxy. Este sistema podrá ser utilizado fácilmente en lugares públicos o determinados eventos.

[1] Un Smartphone es un teléfono celular que permite al usuario conectarse a Internet, dispone de funciones multimedia y GPS entre otras cosas. [80]

[2] Android es una plataforma software para dispositivos móviles desarrollada por Google y diseñado para dispositivos móviles con pantalla táctil, como teléfonos inteligentes, tabletas etc. [81]

1.3. Objetivos

HAPIS es un proyecto social que tiene un objetivo muy claro: dar la oportunidad a personas con recursos a ayudar a otras personas, necesitadas, que están pasando por una mala situación, consiguiendo una conexión real entre estas personas.

Con esta aplicación, se pretende alcanzar los siguientes objetivos:

- Reducir la privación material severa[3] entre los usuarios de la aplicación.
- Proveer un nuevo sistema de ayuda a las personas sin recursos.
- Fomentar el uso de nuevas tecnologías en situaciones cotidianas.
- Hacer más rápido y eficaz el servicio de entrega de productos básicos de ayuda a las personas que lo necesitan sin necesidad de intervención por parte de terceras instituciones.
- Fomentar la relación entre los dos tipos principales de personas, las personas sin recursos y las personas que pueden ayudar. De esta manera, se podría encontrar una nueva modalidad de concienciar a las personas.

1.4. Resumen del trabajo

La idea del proyecto ha surgido porque cada día veo más personas que no pueden llegar al fin del mes y a veces les faltan comida y ropa, que son cosas tan básicas que deberían estar al alcance de cualquier persona.

Cuando he empezado a desarrollar la idea por primera vez, la aplicación era muy diferente del resultado final obtenido. En el periodo de tiempo que llevo trabajando en este proyecto me he dado cuenta que, para obtener un buen resultado, antes de empezar a desarrollar se tienen que tener en cuenta todas las situaciones posibles y los errores y problemas que podrían surgir. Justamente por esto, se necesitan todos los conocimientos adquiridos a lo largo de los años de Universidad.

Por ejemplo, al principio la idea era que las personas necesitadas se creen su propia cuenta y que se comuniquen directamente con los donantes. Pero, al analizar la situación, me he dado cuenta que no todas las personas que necesitan ayuda disponen de un dispositivo móvil conectado a Internet. Además, podría haber muchas cuentas falsas.

Por eso, he decidido que habrá voluntarios que se van a encargar de crear y gestionar las cuentas de estas personas. De esta manera, las cuentas estarían más controladas y las historias de cada persona necesitada podría estar confirmada por el voluntario que la crea. Además, para no implicar ninguna organización, y que todo sea más personal, he pensado que los donantes podrían entregar las donaciones directamente a la persona necesitada.

[3] La privación material severa incluye a aquellas personas que viven en hogares que no pueden afrontar cuatro o más conceptos de consumo, de un total de nueve considerados básicos en el territorio europeo. [82]

O, en el caso que no quieren o no tienen tiempo, el voluntario podría encargarse de la entrega. Por supuesto, teniendo en cuenta que las personas que necesitan ayuda facilitarán su localización temporal. Antes de crear un perfil tendrán que leer la política de privacidad y aceptarla.

En cuanto a la parte extra del Liquid Galaxy, el desarrollo ha ido más lento porque se necesita un sistema funcionando por tal de poder trabajar y debido a la situación actual, que se comentará más adelante, este trabajo ha tenido que ser aplazado. Pero, finalmente he conseguido un sistema y he podido realizar sin problemas esta parte.

Como producto final, he obtenido una aplicación con dos partes principales: una parte a través de la cual se pueden registrar voluntarios o donantes y otra parte, que funciona solamente si el usuario dispone de un sistema Liquid Galaxy, pero que podría ser utilizada fácilmente en sitios públicos o eventos especiales y dar así más visibilidad a las personas que necesitan nuestra ayuda.

1.5. Google Summer Of Code

Como ya se ha comentado en el Resumen, una parte de este proyecto se ha desarrollado también para el programa Google Summer Of Code 2020 [1] que se realiza a nivel mundial y en el cual Google ofrece becas (aproximadamente 4000 $) para los estudiantes universitarios por realizar un proyecto tecnológico durante los tres meses de verano (junio, julio y agosto).

En este caso en concreto, he presentado una propuesta en marzo de este año y en abril me han anunciado que he sido seleccionada para desarrollar la aplicación, de código abierto, para el Liquid Galaxy Project. [3]

La primera edición de este programa se ha celebrado en el año 2005 y ha contado con más de 15.000 estudiantes de todo el mundo. En la edición de este año, para la organización de Liquid Galaxy Project han participado un total de 8 estudiantes de varios países como España, Brasil, India, Holanda y Columbia.

1.6. Estructura del documento

Por tal de conseguir los objetivos descritos anteriormente, el proyecto se ha planificado de una manera gradual y se han definido las siguientes tareas y planificación temporal considerando las etapas que se siguen al construir una aplicación móvil.

En el capítulo 2, ***planificación de proyecto***, se realizará un estudio previo de la situación actual tanto en el contexto de la situación económica como de las posibles tecnologías a utilizar. Seguidamente, se identificarán los posibles riesgos que pueden surgir a la hora

de desarrollar el proyecto, intentando aportar soluciones. Finalmente, se presentará la planificación del proyecto, un estudio de presupuesto, se revisarán algunas aplicaciones similares existentes en el mercado y se realizará un análisis de la competencia.

En el siguiente capítulo, haremos una ***descripción de las tecnologías utilizadas***, el entorno de ejecución y las herramientas que vamos a utilizar para un desarrollo correcto de la aplicación.

A continuación, en el capítulo 4, se hará una ***descripción de la funcionalidad y análisis de requerimientos*** donde se determinará que hace la aplicación y se explicarán más a fondo todas las funcionalidades.

En el capítulo 5, se presentará el ***diseño de la aplicación*** donde podremos ver las diferentes pantallas, explicadas una por una.

Seguidamente, en el capítulo 6, entraremos más a fondo en los detalles del ***desarrollo*** y la parte técnica del proyecto.

Finalmente, en el capítulo 7, ***conclusiones y posibles mejoras,*** se hará una valoración del proyecto y se describirán posibles mejoras futuras.

2. Planificación del proyecto

2.1. Determinación de la situación actual

La problemática a la cual este proyecto ha tratado de dar solución es la situación de pobreza o exclusión social que actualmente sufren muchas personas o familias.

Según el 9º Informe AROPE[4] del 2019, publicado en el año 2020, solo en España, en el año 2018, 12.2 millones personas (26.1% de la población) se encuentran en riesgo de pobreza o exclusión social. Se percibe una leve mejoría en la tasa general pero la pobreza ha aumentado ya que hay colectivos que no perciben dicha mejoría.

A continuación, se mostrará un gráfico con la evolución AROPE en España entre los años 2004 y 2018. [4]

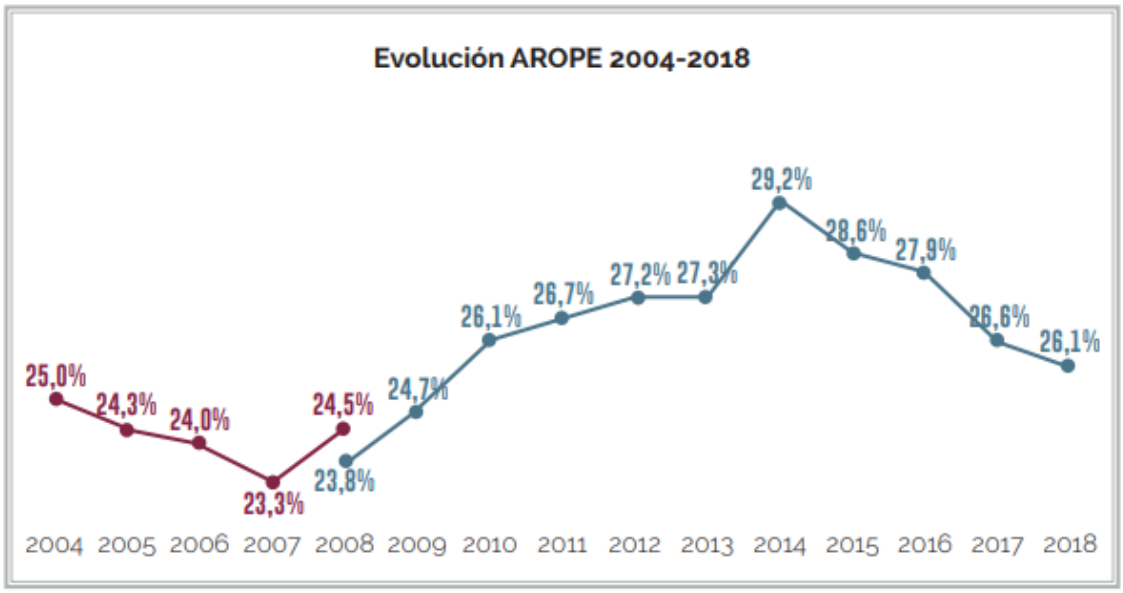


Gráfico 1: Evolución AROPE 2004-2018 España [5]

El gráfico muestra la evolución de la población en riesgo de pobreza y/o exclusión social durante los últimos 14 años en España. En síntesis, puede verse una reducción paulatina de la tasa AROPE hasta poco antes del comienzo de la crisis. A partir del año 2007 se produce un acelerado crecimiento que, incluso con el cambio metodológico, continúa hasta el año 2014 en el que llega a su máximo histórico. Finalmente, en estos cuatro últimos años el cambio de tendencia parece consolidarse, aunque con una reducción más pausada.

En 2018, el 55,3% de la población tiene alguna clase de dificultad para llegar a fin de mes. Este último año la tasa se ha incrementado en dos puntos porcentuales, con lo cual

[4] AROPE - At risk of poverty and / or exclusion

se ha perdido una parte de lo recuperado el año pasado. Además, debe resaltarse que el incremento se ha registrado entre los grupos que experimentan mayor dificultad.

En términos comparativos, la tasa total es la segunda más baja de la década; sin embargo, no debe olvidarse que ello significa que más de la mitad de la población vive en el límite de sus posibilidades, y que algo más de una cuarta parte del total (27,1%) llega a fin de mes con dificultad o con mucha dificultad. En otras palabras y en lo que se refiere a esta variable, se ha progresado, pero esta mejora es absolutamente insuficiente. [5]

Otro concepto que se ha de tener en cuenta es la privación material severa. La PMS[5] que viven en hogares que no pueden afrontar cuatro o más conceptos de consumo, de un total de nueve considerados básicos en el territorio europeo. La privación material severa contabiliza a las personas que: [5]

- No pueden permitirse una comida de carne, pollo o pescado al menos cada dos días.
- No pueden permitirse mantener la vivienda con una temperatura adecuada.
- No tienen capacidad para afrontar gastos imprevistos.
- Han tenido retrasos en el pago de los gastos relacionados con la vivienda principal (hipoteca o alquiler, recibos de gas, comunidad ...) en los últimos 12 meses.
- No pueden permitirse ir de vacaciones al menos una semana al año.
- No pueden permitirse disponer de un teléfono.
- No pueden permitirse disponer de un televisor.
- No pueden permitirse disponer de una lavadora.
- No pueden permitirse disponer de un automóvil.

La privación material es un indicador de vulnerabilidad grave y cada uno de sus ítems es indispensable para la participación en la sociedad europea.

A continuación, se puede ver un gráfico con la Privación Material Severa en España entre los años 2008- 2018.

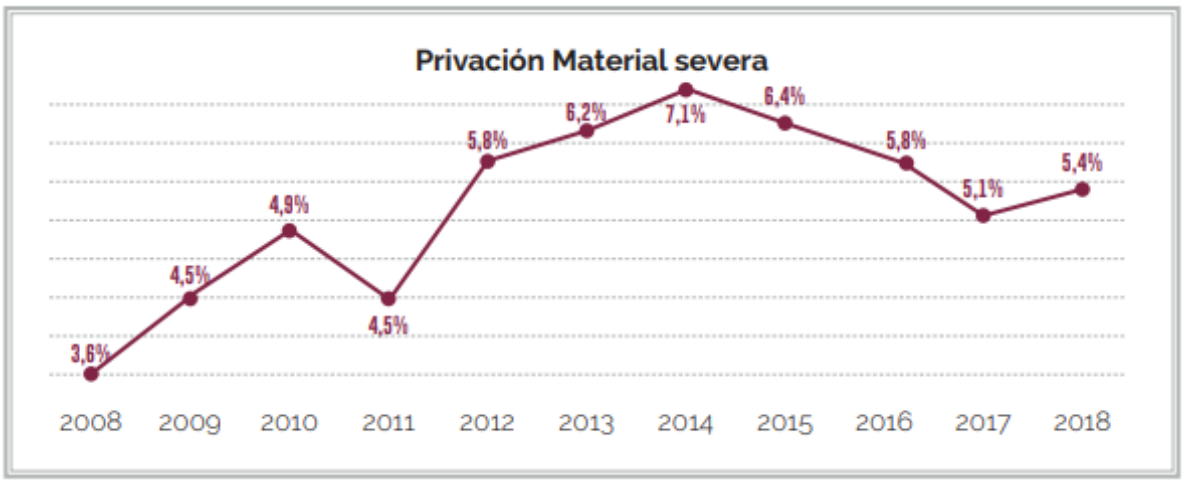


Gráfico 2: Privación Material Severa en España 2008-2018 [5]

[5] La privación material severa

Entre los años 2008 y 2014, la PMS registró un crecimiento casi ininterrumpido y de gran intensidad, que llevó a duplicar su cuantía. A partir del año 2014, la privación material severa comenzó un ciclo de descenso que se mantuvo a razón de seis o siete décimas anuales hasta el año pasado. En el último año, la privación material severa sufrió un pequeño repunte y alcanza al 5,4 % de la población residente en España, lo que supone algo más de 2,5 millones de personas. A pesar de la apreciable reducción experimentada, la PMS aún es muy elevada y llega a unas 850.000 personas más que en el año 2008.

Además, la ONG Oxfam Intermón ha esbozado en su informe "Una reconstrucción justa y necesaria es posible" un panorama muy negro que dejará la Covid-19 este 2020 en el caso de que el PIB descienda un 9%, tal como han estimado diferentes organismos internacionales y españoles, y que el paro pase del 13% de antes de la pandemia al 19%. Si esos pronósticos se cumplen el número de personas que estarían en situación de pobreza relativa se elevaría a 10,8 millones, 700.000 más que en la etapa anterior al coronavirus. [6]

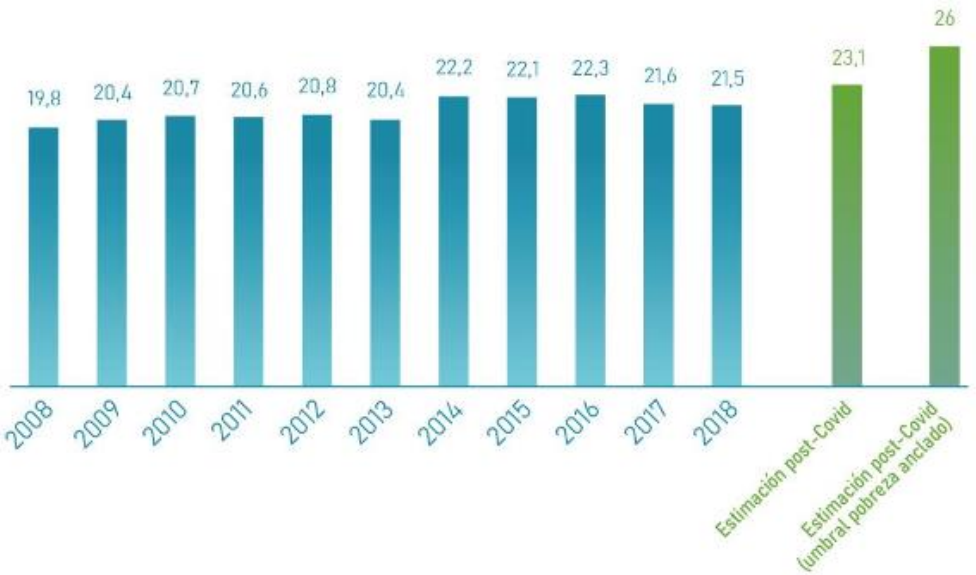


Gráfico 3. Evolución y previsible impacto de la COVID-19sobre la pobreza relativa (en % sobre la población) [6]

Conforme con todo lo que se ha presentado anteriormente, mediante la aplicación móvil desarrollada se tratará de relacionar dos grupos de personas:

- Personas con recursos dispuestas a colaborar por tal de mejorar la calidad de vida de personas que se encuentran en una situación de pobreza
- Personas sin recursos que no disponen de recursos básicos y por tanto necesitan ayuda

En los siguientes apartados se comentará más en detalle la parte relacionada con los perfiles de usuario, pero ya se puede observar que la mayoría de personas sin recursos no siempre disponen de un teléfono móvil con acceso a Internet, por lo que se necesita la ayuda de un voluntario.

Por último, se remarca que se ha considerado importante enfocar este proyecto a un suceso que afecta a muchas personas y que podría llegar a tener repercusiones positivas a nivel social en el caso que tuviera éxito.

2.2. Identificación de riesgos

A la hora de desarrollar un proyecto, en un primer punto se tiene que considerar la gestión de riesgos que pueden afectar al correcto desarrollo del proyecto. La identificación de riesgos y su posterior gestión es uno de los aspectos más importantes para poder mantener el control de un proyecto. Esto nos permitirá anticipar aquellas situaciones que pueden comprometer (o favorecer) los objetivos, y definir de antemano planes de actuación para ellas.

El riesgo se define como una situación conocida, que puede ocurrir o no, y en caso de que ocurra, afectará a nuestra capacidad para cumplir los objetivos del proyecto. Según el tipo de riesgo, hay 2 tipos: [7]

- ***Riesgos del proyecto:*** afecta a la planificación temporal y al coste del proyecto o identifica problemas potenciales de presupuesto o recursos.

- ***Riesgos técnicos:*** amenaza la calidad y planificación temporal del software que hay que producir. Identifica posibles problemas de diseño, implementación, interfaz, verificación y mantenimiento.

La gestión continuada de los riesgos permite aumentar su eficacia:

- Evaluar continuamente que puede ir mal
- Determinar qué riesgos son importantes o implementar estrategias para resolverlos
- Asegurar la eficacia de las estrategias

La identificación de riesgos constituye un intento sistemático para especificar las amenazas al plan del proyecto. Algunos de los tipos de riesgos que pueden aparecer a lo largo de este proyecto son los siguientes:

- Riesgos de tecnología
- Riesgos de personal
- Riesgos organizacionales
- Riesgos de herramientas de desarrollo
- Riesgos de requerimientos
- Riesgos de estimación

En el caso del proyecto que se va a desarrollar se han identificado los siguientes riesgos:

Riesgo	Descripción	Tipo de riesgo
Poca experiencia en las tecnologías utilizadas	Falta de experiencia del desarrollador en el lenguaje de programación utilizado y en la tecnología extra del Liquid Galaxy	Personal y tecnología
Poca experiencia en la planificación del proyecto	Falta de experiencia en la planificación de un proyecto y en la estimación de los tiempos requeridos para el desarrollo del proyecto	Estimación
Tiempo de entrega	Tiempo limitado para la realización del proyecto	Estimación
Problemas para trabajar con la tecnología	Debido a la situación actual relacionada con la salud, la tecnología que se necesita es más difícil de conseguir	Estimación y tecnología

Tabla 1: Identificación de riesgos del proyecto

2.3. Análisis de riesgos

Una vez ya tenemos identificados los riesgos analizaremos la probabilidad que tiene cada riesgo de ocurrir, las consecuencias de los problemas asociados con el riesgo y el impacto que tendrá en la realización del proyecto.

Según la probabilidad que tiene un riesgo de hacerse realidad, tenemos 5 categorías:

- Muy baja
- Baja
- Media
- Alta
- Muy alta

Según el impacte que tiene el riesgo en el proyecto, tenemos 4 categorías:

- Despreciable
- Marginal
- Critico
- Catastrófico

En la siguiente tabla podemos observar un análisis de riesgos del proyecto siguiendo lo que se ha mencionado anteriormente:

Riesgo	Consecuencia	Probabilidad	Impacto
Poca experiencia en las tecnologías utilizadas	Una implementación no correcta de la aplicación, lo que podría llevar a un mal funcionamiento de la misma	Alta	Alto
Poca experiencia en la planificación del proyecto	Tardar demasiado en la realización de cada tarea u organizar mal el trabajo, conlleva a una tardanza en el tiempo de entrega	Media	Alto
Tiempo de entrega	Las tareas tienen que ser muy bien planificadas para evitar una entrega no completa del proyecto	Alta	Alto
Problemas para trabajar con la tecnología	No poder acceder a las tecnologías necesarias para un correcto desarrollo de la aplicación conlleva a una tardanza en el tiempo de entrega	Media	Alto

Tabla 2: Análisis de riesgos

2.4. Proyección de riesgos

En este apartado realizaremos una planificación de los riesgos, fijando las medidas para prevenir, minimizar y contener el riesgo en caso que ocurriera.

A continuación, se describirán las medidas necesarias en cada uno de los casos:

Riesgo: Poca experiencia en las tecnologías utilizadas

- ***Prevención:*** Estudiar, consultar manuales y practicar con las tecnologías que se van a utilizar antes de empezar con el desarrollo del proyecto.

- ***Minimizar el riesgo:*** Consultar tutoriales, más manuales o documentación al largo de la realización del proyecto.

- ***Plano de contingencia:*** En caso de no avanzar, consultar lo necesario con un experto.

Riesgo: Poca experiencia en la planificación del proyecto

- ***Prevención:*** Consultar y analizar documentación y técnicas para la planificación de un proyecto y aumentar el tiempo inicialmente previsto para poder cubrir imprevistos.

- ***Minimizar el riesgo:*** Cumplir siempre con la planificación de cada fase y revisar siempre la planificación inicial del proyecto.

- ***Plano de contingencia:*** En el caso de un desvío en la planificación de alguna fase, volver a planificar las siguientes fases por tal de acabar a tiempo.

Riesgo: Tiempo de entrega

- ***Prevención:*** Realizar una buena planificación del proyecto, tendiendo siempre en cuenta los imprevistos que pueden aparecer al largo del proyecto.

- ***Minimizar el riesgo:*** Revisar periódicamente la planificación del proyecto e ir ajustando la planificación en función de los avances.

- ***Plano de contingencia:*** En el caso de falta de tiempo a la hora de entregar el proyecto, tener en cuenta los requerimientos esenciales para que la aplicación funcione correctamente, quitando requerimientos que no sean imprescindibles.

Riesgo: Problemas para trabajar con la tecnología

- ***Prevención:*** Conseguir toda la tecnología necesaria antes de empezar con el desarrollo del proyecto.

- ***Minimizar el riesgo:*** Buscar otras alternativas temporales en caso que no sea posible acceder a tiempo a las tecnologías necesarias.

- ***Plano de contingencia:*** En el caso que no se consiga probar la aplicación con la tecnología necesaria, implementar solamente las funciones básicas y prescindir de funciones que no sean obligatorias.

2.5. Planificación

Para la planificación del proyecto se ha utilizado la herramienta online Trello. [8]

Trello es una herramienta en línea muy fácil que facilita la colaboración en proyectos. Se pueden añadir ideas, tareas e adjuntar imágenes o enlaces. La herramienta se ha utilizado en general para llevar un control sobre las tares que ya se han finalizado, las tareas en curso y las tares que faltan por hacer.

En la siguiente imagen se puede observar un ejemplo de la herramienta:

Imagen 1: Captura de pantalla de la herramienta Trello

A continuación, se puede observar un diagrama de Gantt, realizada por tal de llevar un mejor control sobre cada fase del proyecto. Se ha realizado una estimación inicial sobre la duración de cada fase, estimadas en número de semanas. De esta manera tendré una ayuda para detectar posibles desviaciones y corregirlas si es necesario. En las siguientes tablas se pueden observar todas las fases del proyecto con la estimación de tiempo para cada una de ellas:

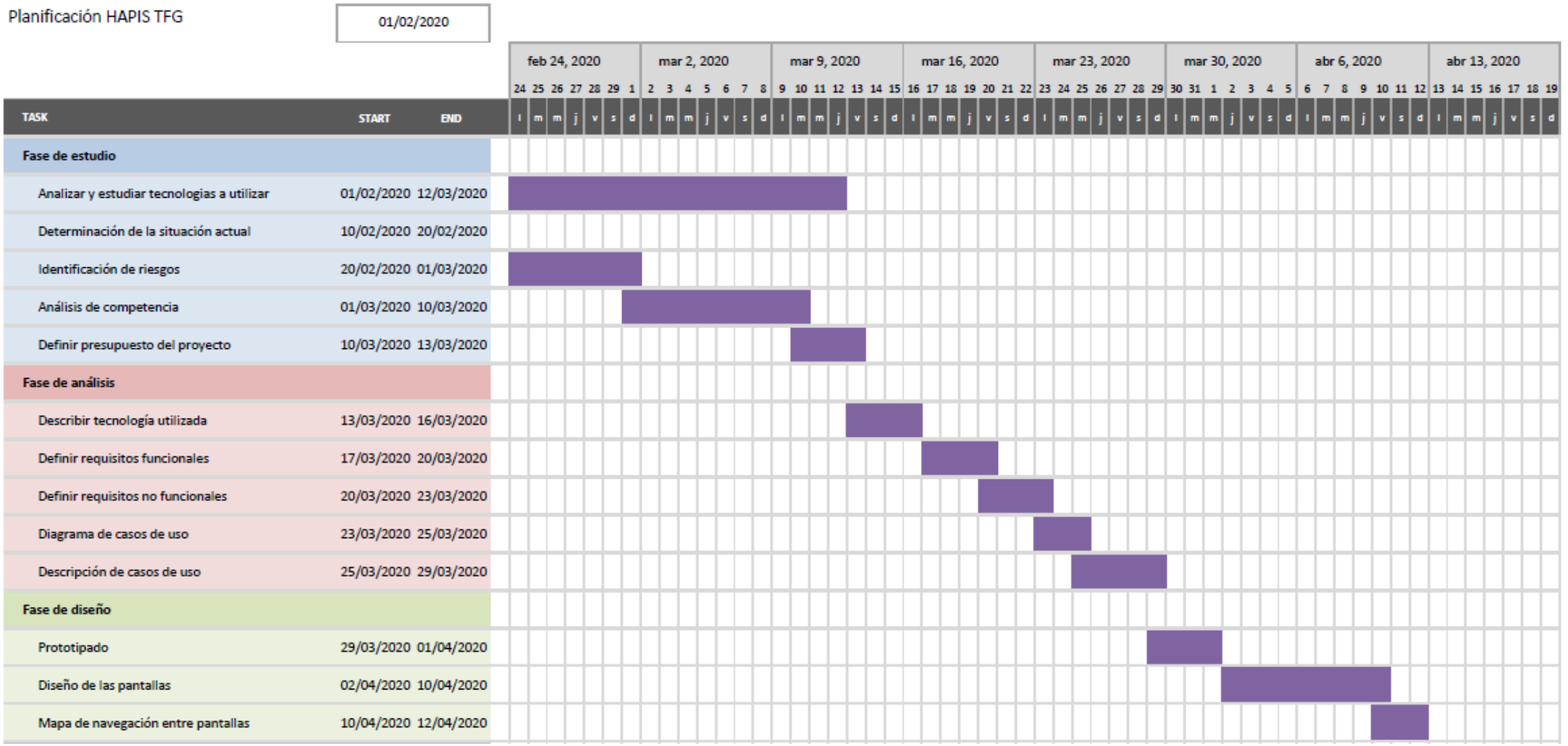


Tabla 3: Planificación temporal del proyecto – parte 1

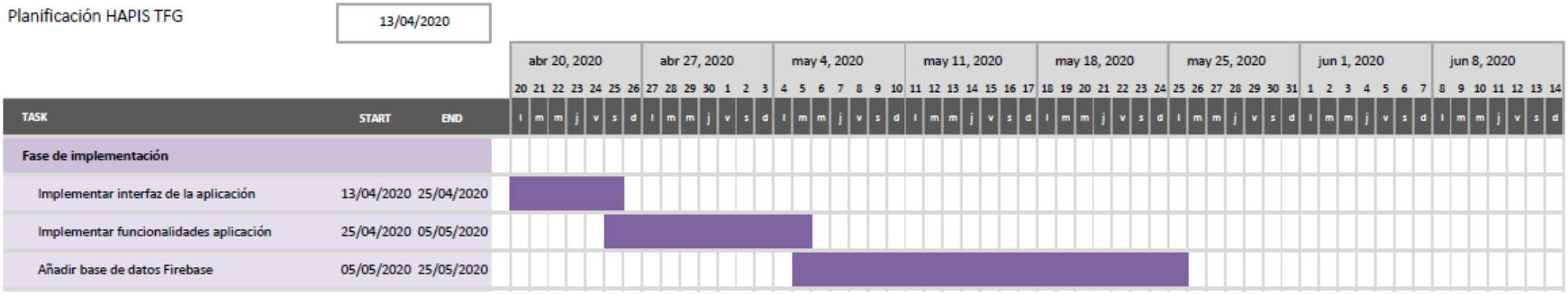


Tabla 4: Planificación temporal del proyecto – parte 2

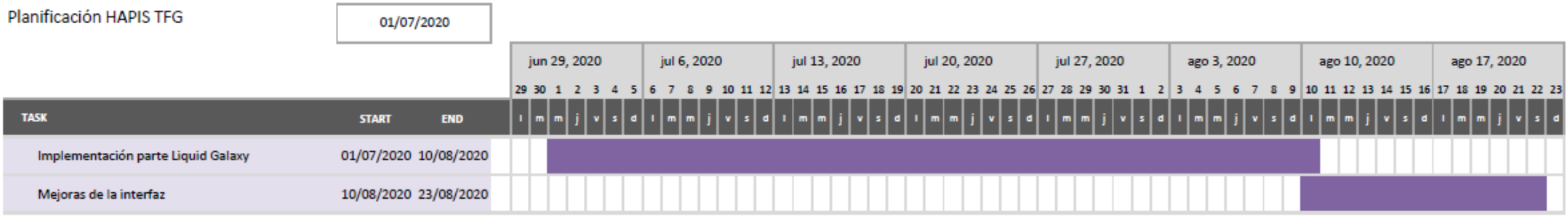


Tabla 5: Planificación temporal del proyecto- parte 3

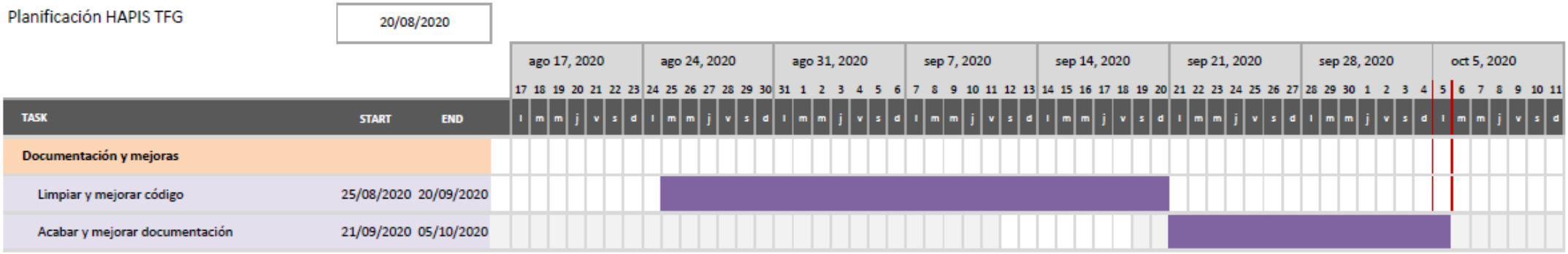


Tabla 6: Planificación temporal del proyecto- parte 4

2.6. Análisis de competencia

El análisis de la competencia consiste en el estudio y análisis de los productos competidores, para que posteriormente, en base a dicho análisis, tomar decisiones estratégicas de cómo enfocar nuestro producto. [9]

2.6.1. Estudio de mercado

En aras de crear una aplicación original y ofrecer nuevos servicios e ideas se realizó una búsqueda de otras aplicaciones similares.

Para este estudio de mercado se ha utilizado la propia tienda online de Android, Google Play, ya que es uno de los mejores lugares donde realizar las búsquedas puesto que da la certeza de que los resultados que se obtienen corresponden a proyectos reales en su fase final de desarrollo o ya finalizados

A continuación, se mostrarán 5 aplicaciones encontradas en el Google Play y una breve descripción de cada una, por tal de poder identificar después las similitudes o diferencias con respecto a la aplicación HAPIS.

Share The Meal: es una aplicación solidaria del Programa Mundial de Alimentos de las Naciones Unidas (WFP) que permite que la gente alimente a niños que lo necesiten. Es una aplicación que permite ayudar en la alimentación de un niño donando dinero a partir de 40 céntimos al día. [10]

Charity Miles: es una aplicación que permite a los usuarios donar dinero para la caridad mientras camina, corre o monta en bicicleta. Su funcionamiento es muy sencillo. Después de descargar la aplicación, hay que elegir la obra de caridad y a lo que se quiere destinar el dinero. A continuación, solo hay que empezar a quemar calorías. Por cada milla que se hace en bicicleta, se donan 10 centavos, y si está corriendo o caminando, 25 centavos. [11]

Olio: Olio fue lanzado a finales de 2015 por Tessa Cook y Saasha Celestial-One. Su funcionamiento es simple: la aplicación conecta a las comunidades locales, negocios y tiendas de alimentos con los usuarios, quienes suben una foto y una descripción de los alimentos que ya no quieren, una respuesta, y una hora y un lugar de selección para que otros vengan a recogerlos. [12]

OurCalling: es una aplicación para dar apoyo a personas sin hogar. La aplicación permite al usuario identificar personas sin hogar y encontrar lugares sociales para compartirlos si están disponibles. [13]

Samaritan: esta aplicación existe para revelar las historias con quienes pasamos todos los días (la madre soltera en el autobús, el abuelo en la calle o la adolescente sin hogar fuera del supermercado) y nos ayuda a responder bien en estos momentos.

Cuando el usuario pasa al lado de un titular de Bluetooth Beacon[6], aparecerá una notificación en su teléfono. Al hacer clic en la notificación, se brinda la oportunidad de leer sobre la historia de la persona o contribuir con 1 dólar o más para un bien o servicio necesario (como alimentos, combustible o ropa).

La aplicación actualmente es compatible solamente con Beacon Holders en Seattle, WA y no se puede descargar en nuestro país. [14]

2.6.2. Conclusiones

Como conclusión a las aplicaciones presentadas anteriormente, se considera importante destacar que no existe una gran variedad de aplicaciones similares, sobretodo en España, la mayoría encontrándose en Estados Unidos. Entre las cinco aplicaciones móviles presentadas, hay una de ellas que ha llamado mi atención debido a la gran similitud con la idea que tengo sobre el proyecto.

La aplicación es Samaritan que genera una notificación cuando el usuario pasa cerca de una persona necesitada que dispone de un dispositivo bluetooth. El usuario puede donar dinero a través de la aplicación y la persona necesitada puede utilizar el dispositivo o una tarjeta facilitada por la organización para comprar artículos de primera necesidad.

Como idea para incluir en un futuro al proyecto, me gustaría valorar e investigar cómo funciona el Beacon Bluetooth, ya que me parece una gran idea y creo que podría ayudar a muchas personas. Por otro lado, la experiencia de usuario sería mucho más simple y flexible utilizando esta tecnología, siempre teniendo en cuenta la privacidad de los usuarios y personas implicadas.

Para acabar, considero que el mercado de las aplicaciones móviles que permiten ayudar a personas sin recursos está muy poco desarrollado, sobretodo en España. Sin embargo, creo que si se publicara una aplicación de este tipo los usuarios se animarían a utilizarla y ayudar a los que más lo necesitan.

[6] Un beacon es un dispositivo basado en tecnología Bluetooth Low Energy (BLE) que está constantemente emitiendo una señal única. Ésta puede ser recibida e interpretada por otros dispositivos, como smartphones y tabletas, que pasen a una distancia corta. Es necesario que el dispositivo receptor tenga el Bluetooth activado y disponga de una app específica para reconocer la señal. [83]

2.6.3. Planificación estratégica de tipo DAFO

La planificación estratégica de tipo DAFO[7] es una herramienta o técnica de planificación estratégica utilizada para identificar las fortalezas, oportunidades, debilidades y amenazas en el desarrollo y publicación de la aplicación que se desarrollará. El DAFO es un elemento fundamental para establecer la base de un buen plan de marketing estratégico. [15]

El objetivo del análisis DAFO es identificar las líneas de acción y planes estratégicos que son necesarios para alcanzar los objetivos propuesto para que la aplicación sea un éxito.

En la tabla que se muestra a continuación se puede observar un análisis DAFO de nuestra aplicación, teniendo identificadas las amenazas, las debilidades, las oportunidades y las fortalezas.

DEBILIDADES	**FORTALEZAS**
▪ Solamente disponible en plataforma Android ▪ Falta de experiencia en el mercado ▪ Depende de voluntarios ▪ Beneficios obtenidos mínimos	▪ Producto novedoso ▪ Ofrece motivación al usuario, ya que a través de la aplicación tendrá la oportunidad de ayudar a alguien ▪ Utilización de buenas prácticas para obtener un buen rendimiento de la aplicación y del dispositivo ▪ Fiabilidad de las tecnologías que se utilizaran (ejemplo: Google Firebase) ▪ Funcionalidad extra del Liquid Galaxy
AMENAZAS	**OPORTUNIDADAES**
▪ Elevado ritmo de innovación en el sector de las aplicaciones móviles ▪ Creación de una aplicación similar por parte de una empresa con más poder en el mercado ▪ Dificultades para encontrar voluntarios dispuestos a ayudar	• Distribución sencilla y de bajo coste • Tendencia creciente de usuarios que descargan este tipo de aplicaciones • No se necesita una gran infraestructura • Posibilidad de expandir y colaborar con varias organizaciones que se dedican a ayudar a las personas

Tabla 7: Análisis de tipo DAFO

Como se puede observar en la planificación estratégica de tipo DAFO, habría que encontrar soluciones sencillas a la hora de gestionar los recursos que el usuario donaría, ya que en gran parte la aplicación depende de voluntarios.

[7] DAFO es una sigla y es la unión de las palabras Debilidades, Amenazas, Fortalezas y Oportunidades

Después de probar la aplicación unos cuantos meses, si se observa una tendencia de expansión habría que pensar en desarrollar también la versión para el sistema operativo iOS[8], ya que así tendríamos una oportunidad de aumentar el número de usuarios.

En cuanto a la implementación de la aplicación, no debería haber problemas y tampoco costes muy elevados, por lo menos al principio.

2.6.4. Plan de marketing

El plan de marketing o plan de comercialización señala las principales estrategias de marketing o estrategias comerciales que se utilizarán para atender el mercado objetivo.

La aplicación estará disponible para descargar en Google Play.

En este caso, el plan inicial sería lanzar la aplicación con un modelo de pago gratuito (free), disponible para la plataforma Android. Aunque la aplicación será disponible de forma gratuita, para obtener beneficios y ayudar al desarrollador con la investigación y el mantenimiento de la aplicación se añadirán dos opciones:

La opción de que el usuario pueda hacer sus propias donaciones a través de un sistema de pago seguro y la opción de que el usuario mire un video de no más de 30 segundos para que la empresa contratante done una cantidad de dinero. Por ejemplo, por cada visualización se donarán 10 céntimos.

También se podría considerar la opción de crear una versión Pro de la aplicación para los usuarios que quieren utilizarla con el sistema Liquid Galaxy, pagando una cantidad mensual o anual para utilizar esta funcionalidad.

Después de lanzar la aplicación, se tiene la intención de realizar una campaña publicitaria en redes sociales, medios especializados o incluso flyers con información por tal de hacer la aplicación conocida.

Como parte de las funcionalidades que se podrían incluir en el futuro, sería la opción de compartir la historia de las personas necesitadas a través del dispositivo Bluetooth Beacon tal y como se ha mencionado anteriormente en el estudio del mercado. Si se llega a implementar esta funcionalidad, se podría utilizar para obtener beneficios económicos, reteniendo un mínimo porcentaje de la donación con fines de investigación y mantenimiento de la aplicación.

[8] **iOS** es un sistema operativo móvil de la multinacional Apple Inc. Originalmente desarrollado para el iPhone (iPhone OS), después se ha usado en dispositivos como el iPod touch y el iPad. [84]

2.7. Presupuesto

Para realizar este proyecto, el coste se dividirá en tres campos a tener en cuenta: Coste de personal, Coste de Software y Coste de Hardware.

2.7.1. Coste de personal

Para calcular el coste del personal en caso de este proyecto, se han tenido en cuenta los costes medios de los trabajadores en España en las diferentes áreas.

Puesto de trabajo	Número de horas	Coste por hora	Coste total (euros)
Programador	600	35	21000
Analista	120	20	2400
Redactor	80	15	1200
			Total: 24600

Tabla 8: Calculo del coste de personal para el proyecto

2.7.2. Coste de Software

No habrá coste alguno en el desarrollo del proyecto. Se desarrollará de forma abierta utilizando el programa de desarrollo Android Studio, el repositorio abierto de GitHub, la tecnología de código abierto Liquid Galaxy y la base de datos de Firebase, haciendo uso del plan gratuito para la gestión de los datos.

2.7.3. Coste de Hardware

En este caso se hará uso del ordenador personal y para la tecnología Liquid Galaxy el Lleida Liquid Galaxy LAB me ha prestado tres ordenadores, tres pantallas y una Tablet para poder realizar todas las pruebas necesarias. A continuación, se encuentra una estimación del coste de todo el equipamiento utilizado:

Equipamiento	Coste(euros)
1 ordenador portátil Lenovo	500
3 ordenadores Gigabyte i7	3x550
3 monitores Benq GL2450	3x100
Tablet Acer B3-A40	160
Teléfono móvil Samsung Galaxy A40	220
	Total: 2830

Tabla 9: Estimación de coste hardware

2.8. Metodología de desarrollo

Por tal de conseguir los objetivos descritos anteriormente, el proyecto se ha planificado de una manera gradual y se ha utilizado como metodología de trabajo el modelo de Proceso de la Ingeniería de la Usabilidad y la Accesibilidad[9] , incluido en el Diseño Centrado en el Usuario, ya que se enfoca en los usuarios, en la experiencia y su grado de satisfacción.

2.8.1. Modelo de Proceso de la Ingeniería de la Usabilidad y la Accesibilidad

MPIu+A es una metodología de desarrollo de sistemas interactivos que siguen los principios del Diseño Centrado en el Usuario cuyas características principales son: [16]

Organización Conceptual

Una de las metas más importantes de este modelo es intentar reunir el modelo de desarrollo de sistemas interactivos de la Ingeniería del Software con los principios básicos de la Ingeniería de la Usabilidad y los de la accesibilidad proporcionando una metodología que sea capaz de guiar a los equipos de desarrollo durante el proceso de implementación de un determinado sistema interactivo.

Tiene tres pilares básicos

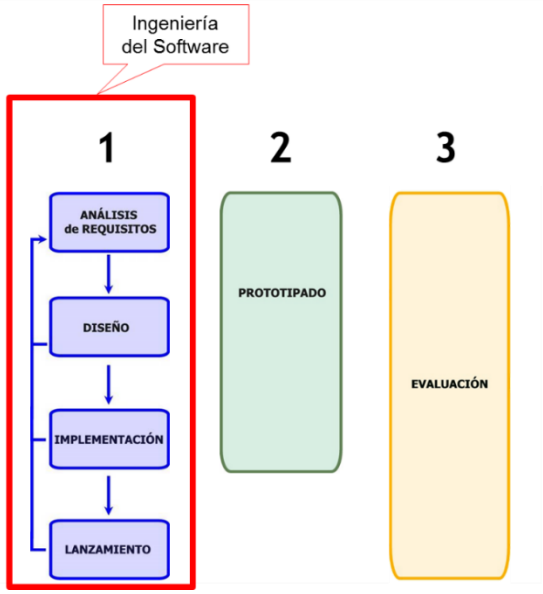


Imagen 2: Pilares modelo MPIu+a [16]

[9] MPIu+a = Modelo de Proceso de la Ingeniería de la Usabilidad y la Accesibilidad. [16]

En la imagen anterior se pueden observar tres conceptos principales:

- ***La Ingeniería de Software:*** en el formato "clásico" de ciclo de vida en cascada iterativo o evolutivo
- ***El prototipado:*** como metodología que engloba técnicas que permitan la posterior fase de evaluación.
- ***La evaluación:*** engloba y categoriza los métodos de evaluación existentes.

El usuario

En los modelos de desarrollo actuales los diseñadores o los programadores normalmente escogen por el usuario, pero para que una aplicación sea realmente "amigable con el usuario" *(user-friendly)*, este debería calificarla, ya que es la persona que más interacciona con el sistema.

Un proceso de Diseño Centrado en el Usuario debe dejar claro que el usuario está en la parte central y por encima del resto de etapas de todo el modelo de proceso.

En la siguiente imagen se puede observar:

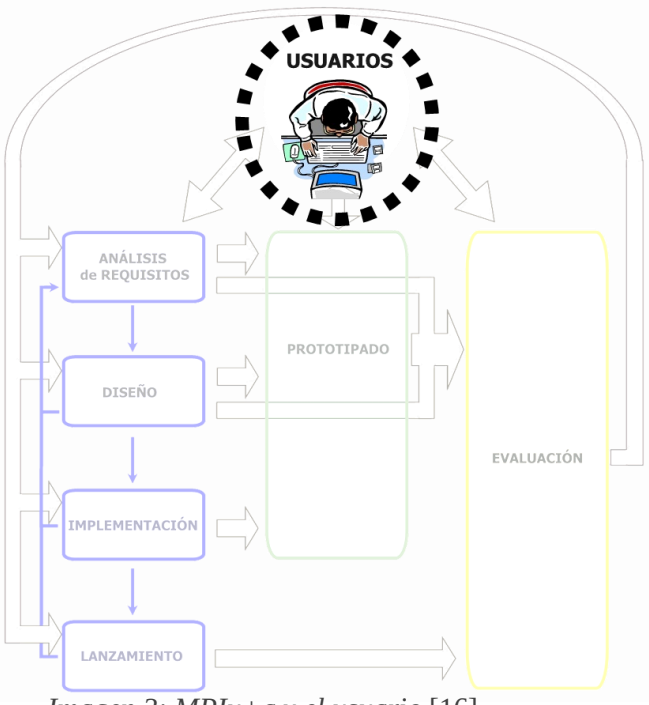


Imagen 3: MPIu+a y el usuario [16]

Otro aspecto determinante en este modelo de proceso es que se da mucha importancia no sólo a los usuarios, sino también a los implicados (o stakeholders) en cuanto a que son personas que sin ser usuarios directos del sistema su actividad se ve afectada por el mismo.

Es un método iterativo

El proceso de repetición existe en este modelo y se produce a lo largo del proceso de desarrollo.

El esquema propuesto a continuación dispone de una serie de flechas cuyo objetivo no es otro que visualizar que desde todas las fases se promueve la participación activa de los usuarios, tanto en el análisis de requisitos como en el diseño y en la realización de prototipos y/o su posterior evaluación.

Pueden observarse dos tipos de flechas, unas delgadas que se corresponden con el modelo de la Ingeniería del Software, y otras más gruesas que convierten la Ingeniería del Software en un verdadero modelo centrado en el usuario. Éstas últimas indican, entre otras cosas, donde interviene el usuario.

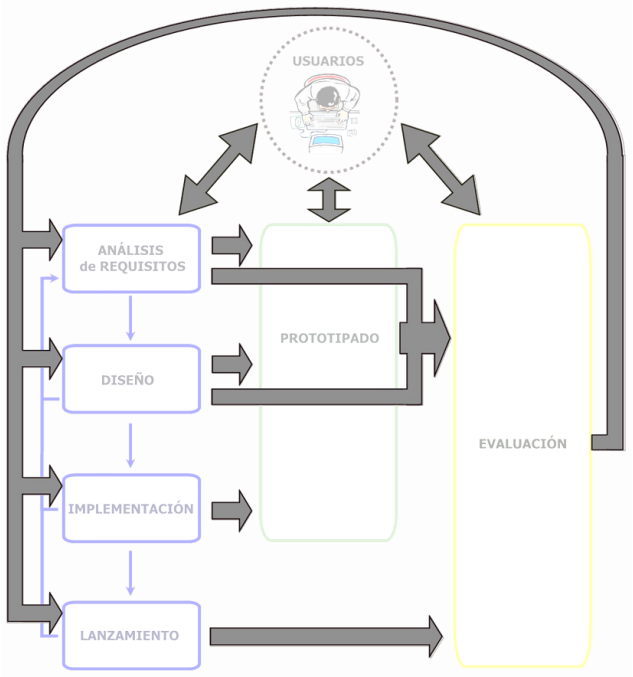


Imagen 4: MPIu+a e iteratividad [16]

Sencillez

Las interfaces de los sistemas interactivos tienen que ser sencillas y simples, pero sin perder su capacidad comunicativa y funcional.

La metodología que permita a los desarrolladores llevar a cabo su trabajo de manera más eficiente tiene que ser también muy sencilla y simple.

El esquema aquí presentado nace con esta idea como trasfondo para todo el equipo de desarrollo: Es escueto, con pocos nodos y ramificaciones y sin caminos condicionales que dificultan su comprensión.

Flexibilidad

Se tiene que considerar el hecho de que el modelo no tiene ni un sentido lineal ni restrictivo, sino que fomenta la libre aplicación del mismo, ya que será el desarrollador junto con los propios requisitos del sistema, las particularidades de los usuarios y los resultados de las diferentes evaluaciones quien marcará cuantas iteraciones deben realizare, como deben hacerse y el flujo de las acciones a realizar en cada iteración.

2.8.2. Pruebas con usuarios

Tal como se ha comentado anteriormente, la aplicación se basa en el modelo centrado en el usuario, teniendo en cuenta que se ha incluido a varios usuarios activamente para realizar pruebas y dar su opinión sobre el diseño y las funcionalidades de la aplicación a lo largo del proyecto de desarrollo.

El propósito principal de las pruebas ha sido el de analizar la usabilidad y el correcto funcionamiento de la aplicación.

Las pruebas se han realizado utilizando un archivo. apk de la aplicación que ha sido enviado a cada uno de los participantes junto con un formulario con diferentes preguntas sobre las funcionalidades y el diseño de la aplicación.

Después de tener todas las respuestas, se han analizado y se han realizado los cambios que consideraba más importantes y las cosas que han destacado más participantes.

Las pruebas se han realizado varias veces, en diferentes fases del proceso.

Durante las pruebas se ha recolectado la siguiente información:

- Nivel de dificultad para entender y utilizar la aplicación a nivel de Interfaz
- Errores encontrados al utilizar las funcionalidades
- Mejoras y funciones sugeridas

Estudiando la información recolectada, se pueden mencionar las siguientes observaciones realizadas por la mayoría de los participantes:

- En el perfil del donante, el botón del menú lateral no funciona.
- En el mapa, si seleccionas un marcador de una persona no se realiza ninguna acción.
- En el perfil del voluntario, al editar la información de un cuadro de texto, se borra la información anterior.
- La interfaz en general, bien diseñada y entendedora.
- Algunas descripciones de las personas necesitadas demasiado largas.

Participantes

Durante el proceso se han utilizado un total de 16 usuarios, seleccionados en función de su edad. 7 de los usuarios tenían entre 17 y 23 años, 6 de los usuarios entre 23 y 35 años y 3 usuarios mayores de 35 años.

En este caso, los participantes han sido los estudiantes de los laboratorios Liquid Galaxy, los mentores de la beca GSoC, algunos estudiantes que colaboran con los laboratorios desde otros países y familiares y conocidos. Hay que mencionar que no ha habido ninguna compensación para ninguno de ellos.

A continuación, se presentan las características de cada perfil:

- ***Perfil de usuario 1 – Personas jóvenes***

  Edad: 17-23

  Género: Masculino y Femenino

  Ubicación: Lleida, Tenerife, Madrid, India, Brasil, Estados Unidos

  Conocimientos tecnológicos: Uso básico de las herramientas de programación

  Uso del teléfono móvil: Entre 2 y 8 horas al día

- ***Perfil de usuario 2 – Personas adultas***

  Edad: 23-35

  Género: Masculino y Femenino

  Ubicación: Lleida, Tenerife, Madrid, India, Brasil, Estados Unidos

  Conocimientos tecnológicos: Uso medio y avanzado de las herramientas de programación

  Uso del teléfono móvil: Entre 1 y 6 horas al día

- ***Perfil de usuario 3 – Personas mayores de 35 años***

  Edad: >35

  Género: Masculino y Femenino

  Ubicación: Lleida, Tenerife, Madrid, India, Brasil, Estados Unidos

  Conocimientos tecnológicos: Uso medio y avanzado de las herramientas de programación

  Uso del teléfono móvil: Entre 1 y 6 horas al día

Los participantes han realizado las pruebas en sus propios dispositivos y el contacto se ha realizado a través de correo electrónico, o en algún caso a través de una videoconferencia.

Cabe destacar que el test se ha realizado de forma no moderada, teniendo en cuenta la ubicación de los participantes.

Como herramientas, para el formulario Pre – Evaluación se ha utilizado un formulario de Google [17] y para las tareas y las preguntas sobre la experiencia de usuario se ha utilizado la herramienta online Loop11. [18]

Loop11 es una gran opción que permite realizar el test en abierto o restringiendo el acceso por IP; o reclutar participantes por perfil (según sexo, rango de edad, nivel de estudios, etc.). Aquí se pueden definir tareas que el usuario debe llevar a cabo o bien preguntas que debe responder. En este caso, se han añadido 12 tareas y como parte de la Post-Evaluación se han añadido dos preguntas.

En la siguiente imagen se puede ver la página principal de la herramienta, en la que se puede consultar información sobre el proyecto, como tiempo medio por tarea, información sobre cada tarea o información general.

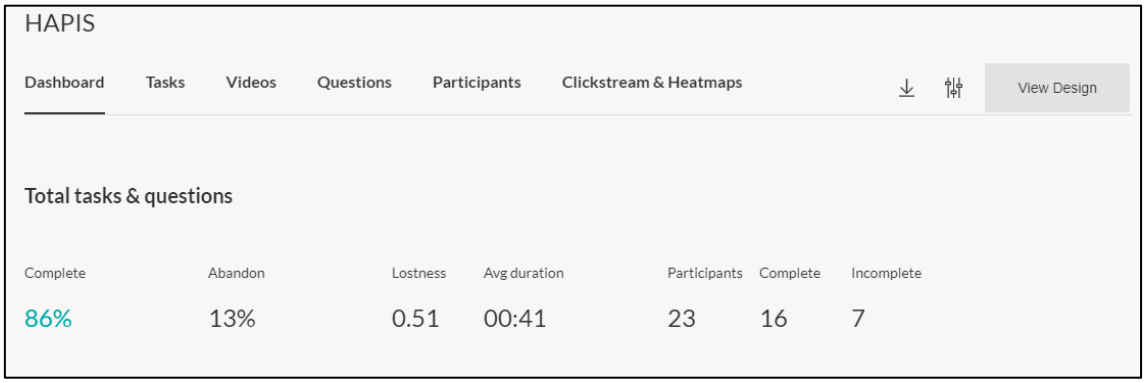


Imagen 5: Dashboard Loop11

Diseño del estudio

El objetivo de las pruebas era probar la aplicación es sus diferentes fases con diversos tipos de usuarios y recolectar la información necesaria para ser capaz de evaluar si la aplicación se está desarrollando de una manera correcta o, al contrario, si se necesita realizar cambios en el diseño o mejorar alguna funcionalidad.

Cada participante ha realizado la prueba recibiendo la misma información con las mismas instrucciones. A continuación, se muestra la información obtenida al finalizar la prueba:

- Numero de tareas llevadas a cabo sin ayuda
- Tiempo para acabar una tarea
- Número y tipo de errores
- Comentarios de los participantes
- Satisfacción con el sistema

Tareas

Con el objetivo de testear la mayor parte de las funcionalidades de la aplicación, se han creado un total de 12 tareas para cada usuario.

A continuación, se muestran las tareas creadas:

- Regístrate como donante
- Inicia sesión como donante y realiza las siguientes operaciones:
    - Realiza una donación personalmente a un usuario que consideras
    - Realiza una donación a través de un voluntario a un usuario que consideras
    - Consulta el mapa de las personas necesitadas
    - Consulta la lista de mapas de las personas necesitadas
    - Realiza una donación para los desarrolladores
- Regístrate como voluntario
- Inicia sesión como voluntario y realiza las siguientes operaciones:
    - Crea un perfil nuevo para una persona necesitada
    - Edita la información para este perfil y cambia la historia de vida
    - Contacta un donante para realizar la entrega de una donación
    - Contacta con nosotros
- Si dispones de un sistema Liquid Galaxy, entra en esa parte de la aplicación y mira la información relacionada con una persona necesitada que escoges

Procedimiento

Las pruebas se han realizado en dos ocasiones, con los mismos participantes. La primera sesión de pruebas se ha llevado a cabo a mitades de julio cuando ya estaban implementadas las funcionalidades básicas del Liquid Galaxy y la segunda sesión de pruebas se ha realizado a finales de agosto cuando el proyecto estaba en una fase final.

En las dos ocasiones se ha seguido el mismo procedimiento:

Después de enviar por correo electrónico toda la información y los archivos necesarios, como el archivo. apk de a aplicación, el link del formulario Pre-Evaluación y el link para acceder a la prueba, los participantes que tenían alguna pregunta extra han sido contactados por videoconferencia y se les ha explicado lo que no entendían. Cada participante ha tenido un tiempo de 3 días para enviar la respuesta por correo electrónico. Cabe destacar algunas de las instrucciones más importantes:

- El objetivo de la prueba es evaluar la aplicación, no al participante
- Cada participante puede decidir libremente el tiempo que utiliza para realizar una tarea
- Durante la prueba, el participante no puede recibir ayuda por parte del autor de la aplicación

Entorno de prueba

Las pruebas se han llevado a cabo en dispositivos móviles que tenían instalado el sistema operativo Android y para la herramienta Loop11, se ha utilizado el ordenador. En cada caso, los dispositivos utilizados salen en la herramienta Loop11. A continuación, se puede ver una imagen:

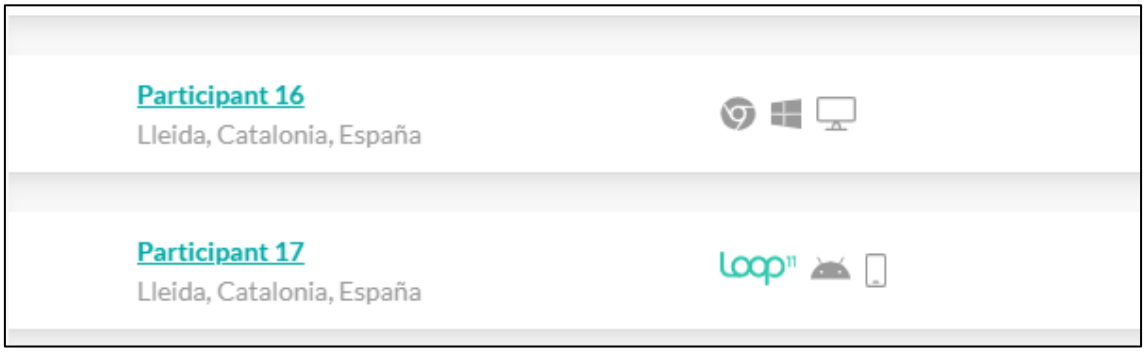


Imagen 6: Información Loop11 - dispositivo y sistema operativo que utiliza cada participante

Documentos y formularios

Durante la prueba se han utilizado los siguientes documentos:

- Un formulario Pre-Evaluación creado con Google Forms. [17]
- Un link a la herramienta Loop11 [18] que contiene las tareas y las preguntas Post-Evaluación

En el Anexo 2 se han adjuntado muestras de cada formulario mencionado anteriormente.

Métricas de usabilidad

El objetivo de realizar estas pruebas fue medir el nivel de usabilidad de todos los usuarios para poder utilizar la aplicación de manera que el nivel de eficacia, eficiencia y la satisfacción obtenida sea lo más alta posible basándonos en la opinión de los usuarios y los resultados de los formularios. A continuación, se describen las métricas obtenidas:

- ***Efectividad:*** calculada a partir de la ratio de tareas resueltas correctamente y tareas fallidas
- ***Eficiencia:*** midiendo el tiempo que los usuarios dedicaron a realizar las tareas asignadas
- ***Satisfacción:*** teniendo en cuenta las opiniones de los usuarios en las preguntas Post-Evaluación

Hay que comentar que, debido a la baja cantidad de participantes, las tres medidas comentadas se han realizado manualmente, teniendo en cuenta los datos obtenidos con la herramienta Loop11.

Puntuación de datos

En esta sección se describen las medidas utilizadas para obtener los resultados.

1. ***Efectividad***

La tarea se realiza con éxito si el participante obtiene el resultado adecuado sin ayuda. La métrica se puede calcular para cada tarea siguiendo la siguiente fórmula:

Efectividad de éxito = Número de tareas completadas con éxito / Número total de tareas

La tarea falla si el participante no obtiene el resultado adecuado sin ayuda. Esta métrica se puede calcular para cada tarea siguiendo la siguiente fórmula:

Efectividad de fallos = Número de tareas fallidas / Número total de tareas

Se proporcionará una recopilación de todos los errores en cada tarea, que mostrará la forma en que los usuarios piensan.

2. ***Eficiencia***

Teniendo en cuenta que algunos de los usuarios podrían desviarse al realizar una tarea, los tiempos pueden ser diferentes en cada caso. Recolectando esta información, podemos calcular el tiempo promedio por cada tarea utilizando la siguiente formula:

Tiempo promedio tarea = Suma de todos los tiempos en realizar una tarea / Número total de tareas realizadas

3. ***Satisfacción***

Para evaluar la satisfacción se han considerado las respuestas y opiniones dadas por cada usuario en las preguntas Post-Evaluación.

Resultados

En el Anexo 3 se podrán encontrar los resultados obtenidos para cada tarea, primero en la primera sesión de pruebas y después en la segunda sesión.

Teniendo en cuenta que las pruebas se han realizado con 16 usuarios, los resultados son pocos para obtener unos comentarios más confiables. Pero, de todos modos, la

información es bastante útil para ser utilizada en aplicar las mejoras oportunas en la aplicación.

A continuación, se analizarán los resultados en cada caso:

Efectividad

Teniendo en cuenta que no han sido errores mayores y que la ratio de suceso ha subido en la segunda sesión de pruebas, la efectividad es muy buena.

Eficiencia

El tiempo para realizar una tarea es bastante similar entre los usuarios y se puede observar una mejora de los tiempos en la segunda sesión de pruebas. La tarea que más tiempo ha llevado ha sido la de crear un nuevo perfil para una persona necesitada, pero hay que considerar que tiene varios pasos. De todos modos, teniendo en cuenta que cada tarea debería poder llevarse a cabo en menos de 2 minutos y casi ningún usuario a superado este tiempo, se puede decir que la eficiencia es bastante buena.

Satisfacción

Para evaluar la satisfacción se han considerado únicamente las opiniones de los participantes, los cuales han sido bastante buenas. También hay que remarcar que algunas opiniones han ayudado mucho a la hora de corregir errores en la aplicación, tanto a nivel de diseño como a nivel de funcionalidades.

Mejoras aplicadas

Después de analizar los resultados, se han añadido como mejoras las siguientes:

- Cambiar los iconos de los botones y del texto para poner todos del mismo estilo
- Añadir una acción al seleccionar un marcador en el mapa
- Arreglar el botón del menú lateral en el perfil del donante
- Arreglar el cuadro de texto al editar un perfil de una persona necesitada para que no se borre la información anterior
- Editar tamaño del texto en el Liquid Galaxy al mostrar una ventana con información

3. Descripción de la tecnología utilizada

En este capítulo se van a presentar las principales tecnologías utilizadas y las herramientas que se van a necesitar para llevar a cabo este proyecto con éxito.

3.1. Tecnologías utilizadas

3.1.1. Android

Teniendo en cuenta que la aplicación se publicará en España, se ha realizado un estudio de las tecnologías móviles disponibles en la actualidad en España, donde se ha encontrado entre ellas una que encaja con el objetivo del proyecto debido principalmente a su universalidad. Según un estudio realizado por IAB Spain[10] la penetración de Internet en España en individuos mayores de 14 años alcanza el 82% de los cuales el 96% tiene smartphone y el tiempo medio de consumo al día de Internet desde un dispositivo móvil se sitúa en 2 horas 53 minutos con unas 230 sesiones por individuo al mes (7,6 veces al día). [19]

A continuación, se pueden observar en un gráfico los datos comentados anteriormente:

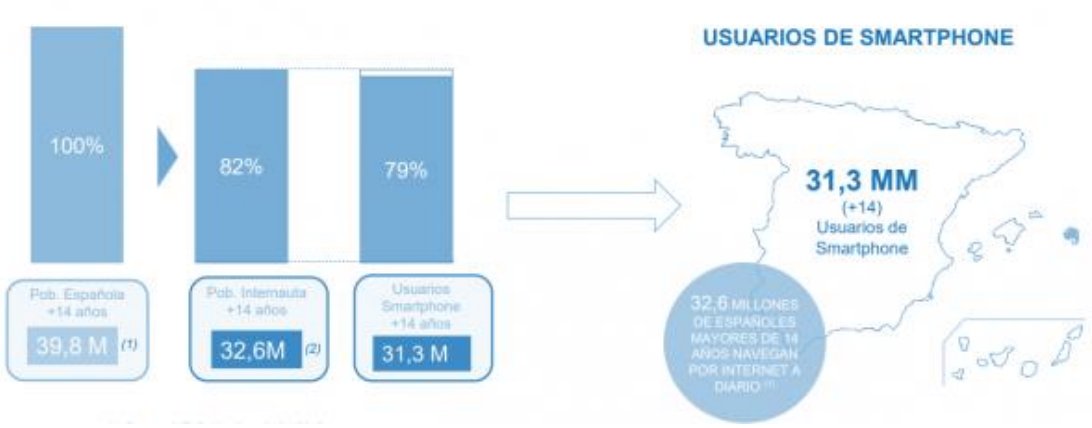


Imagen 7: Penetración usuarios Smartphone España 2019 [19]

Según Kantar, analistas del mercado, Android sigue como líder absoluto entre los sistemas operativos. Google se coloca por encima del 90% en España mientras que iOS cae. Y, aún hay un 0,2% reservado para otros sistemas aún en funcionamiento.

En este mercado, Android crece aún más y ya supera el 90% de cuota. Concretamente, un 90,9%, casi cinco puntos porcentuales más que en el primer trimestre de 2018, cuando el sistema operativo de Google representó el 86,1% de las ventas en el país. iOS, por su parte, se deja casi los mismos cinco puntos por el camino, bajando de un 13,6% en 2018 a un 8,9% en 2019. [20]

[10] IAB Spain es el representante y promotor del sector de la publicidad y la comunicación digital en España

A continuación, se puede observar la cuota de mercado de sistemas operativos Android y iOS en España en el primer trimestre de 2018 y el primer trimestre de 2019:

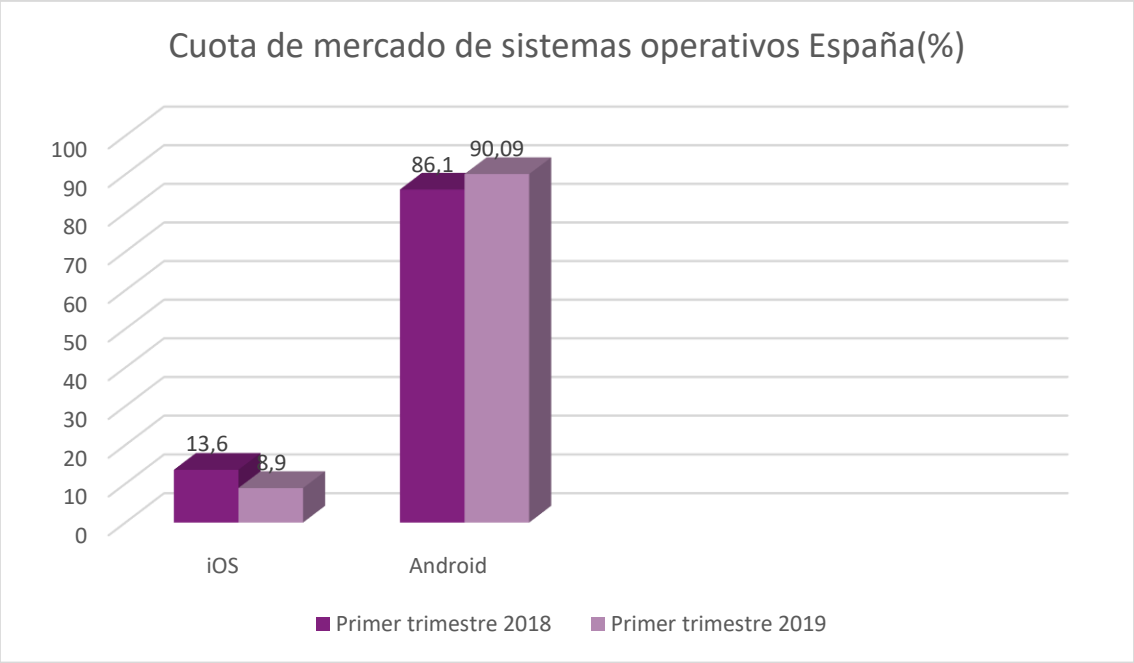


Gráfico 4: Cuota de mercado de sistemas operativos España (%) [20]

Debido al gran apoyo que recibe por parte del mercado, así como por ser de desarrollo libre, unido a todo lo recogido anteriormente en este apartado, Android parece el candidato perfecto para este proyecto.

Tal como se ha comentado anteriormente, Android es un sistema operativo móvil desarrollado por Google. Aunque dispone de interfaces de usuario adaptadas a televisores (Android TV), coches (Android Auto) etc., está especialmente diseñado para dispositivos con entrada táctil como teléfonos o tabletas.

Su kernel está basado en Linux y es un sistema operativo bajo licencia de Apache 2.0, por lo que se considera software libre. Es una plataforma de código abierto, que garantiza los derechos de la libertad en la licencia de usar, compartir, modificar y distribuir en versiones modificadas del software original. No obstante, tiene aspectos controvertidos que ponen en duda si es 100% libre, porque, aunque los servicios de Google son gratuitos, no sabemos el tratamiento de nuestros datos privados. Otro aspecto es la dependencia de los servicios de Google, debido a que es obligatorio iniciar sesión con un ID de Google, el intercambio constante de datos con Google y el envío de la ubicación por defecto. Por contra, parte de los aspectos positivos resultan muy beneficiosos debido a que es de código abierto, tiene una mejor mantenibilidad y seguridad, transparencia en el uso de sensores y dispone de los servicios de Google.

El SDK[11] de Android proporciona todas las herramientas que se necesitan para desarrollar aplicaciones además de la utilización del lenguaje de programación Java.

Android es una pila de software de código abierto basado en Linux creada para una variedad amplia de dispositivos y factores de forma. En el siguiente diagrama, se muestran los componentes principales de la plataforma Android. [21]

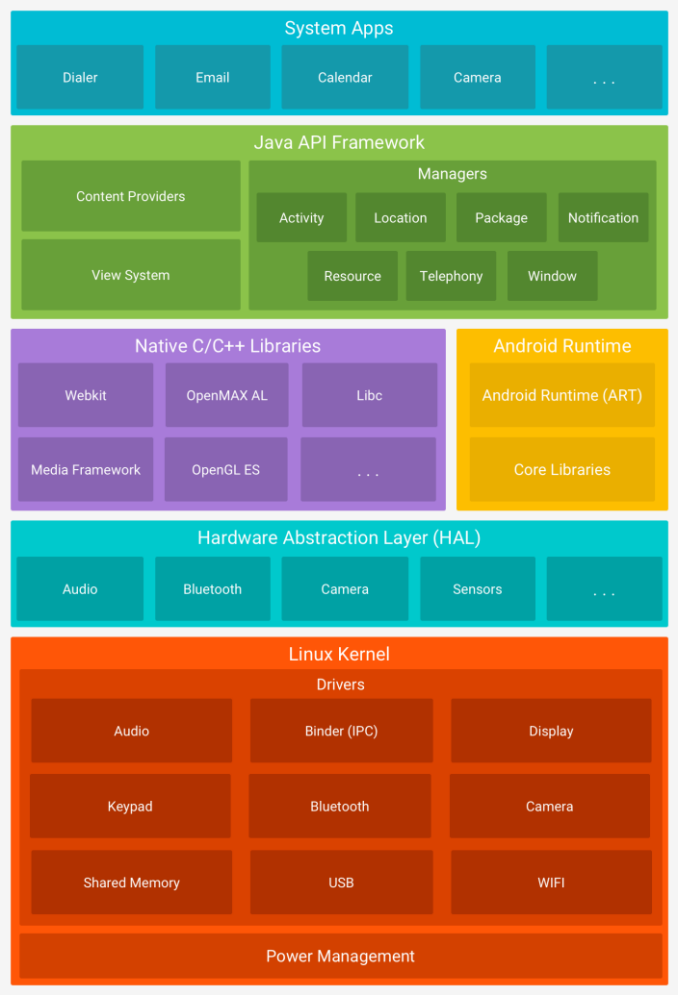


Imagen 8: Pila de software de Android [21]

Como se puede observar, Android tiene una arquitectura de tipo pila, en la cual los componentes se agrupan en capas. Cada una de las capas utiliza elementos de la capa inferior para realizar su función. A continuación, se presentará brevemente cada capa:

- ***Kernel de Linux:*** es la base de la plataforma Android. El uso del kernel en Linux permite que Android aproveche funciones de seguridad claves y, al mismo tiempo, permite a los fabricantes de dispositivos de desarrollar dispositivos controladores de hardware para un kernel conocido.

[11] Software Development Kit

- ***Capa de abstracción de hardware (HAL)****:* brinda interfaces estándares que exponen las capacidades de del dispositivo al marco de trabajo de la API de Java de nivel más alto. Consiste en varios módulos de bibliotecas y cada uno de estos implementa una interfaz para un tipo específico de componente hardware, como el módulo de la cámara o e bluetooth. Cuando el marco de trabajo de una API realiza una llamada para acceder a hardware del dispositivo, el sistema Android carga el módulo de biblioteca para el componente de hardware en cuestión.

- ***Tiempo de ejecución de Android:*** Para los dispositivos con Android 5.0 (nivel de API 21) o versiones posteriores, cada app ejecuta sus propios procesos con sus propias instancias del tiempo de ejecución de Android (ART). El ART está escrito para ejecutar varias máquinas virtuales en dispositivos de memoria baja ejecutando archivos DEX, un formato de código de bytes diseñado especialmente para Android y optimizado para ocupar un espacio de memoria mínimo. Crea cadenas de herramientas, como Jack, y compila fuentes de Java en código de bytes DEX que se pueden ejecutar en la plataforma Android.

- ***Bibliotecas C/C++ nativas:*** Muchos componentes y servicios centrales del sistema Android, como el ART y la HAL, se basan en código nativo que requiere bibliotecas nativas escritas en C y C++. La plataforma Android proporciona API del marco de trabajo de Java para exponer la funcionalidad de algunas de estas bibliotecas nativas a las apps.

- ***Marco de trabajo de la API de Java:*** Todo el conjunto de funciones del SO Android está disponible mediante API escritas en el lenguaje Java. Estas API son los cimientos que necesitas para crear apps de Android simplificando la reutilización de componentes del sistema y servicios centrales y modulares, como los siguientes:
    - Un *sistema de vista* enriquecido y extensible que puedes usar para compilar la IU de una app; se incluyen listas, cuadrículas, cuadros de texto, botones e incluso un navegador web integrable.
    - Un *administrador de recursos* que te brinda acceso a recursos sin código, como strings localizadas, gráficos y archivos de diseño.
    - Un *administrador de notificaciones* que permite que todas las apps muestren alertas personalizadas en la barra de estado.
    - Un *administrador de actividad* que administra el ciclo de vida de las apps y proporciona una pila de retroceso de navegación común.
    - *Proveedores de contenido* que permiten que las apps accedan a datos desde otras apps, como la app de Contactos, o compartan sus propios datos.

Los desarrolladores tienen acceso total a las mismas API del marco de trabajo que usan las apps del sistema Android.

Apps del sistema: En Android se incluye un conjunto de apps centrales para correo electrónico, mensajería SMS, calendarios, navegación en Internet y contactos, entre otros elementos. Las apps incluidas en la plataforma no tienen un estado especial entre las apps

que el usuario elije instalar; por ello, una app externa se puede convertir en el navegador web, el sistema de mensajería SMS o, incluso, el teclado predeterminado del usuario (existen algunas excepciones, como la app Settings del sistema).

Las apps del sistema funcionan como apps para los usuarios y brindan capacidades claves a las cuales los desarrolladores pueden acceder desde sus propias apps.

3.1.2. Firebase

Firebase es una plataforma cloud computing [12] para el desarrollo de aplicación web y móviles desarrollada por Google en 2014. Es una plataforma que se encuentra en la nube, integrada con Google Cloud Platform, que usa un conjunto de herramientas para la creación y sincronización de proyectos que serán dotados de alta calidad, haciendo posible el crecimiento del número de usuarios y dando resultado también a la obtención de una mayor monetización. La plataforma está subida en la nube y está disponible para diferentes plataformas como iOS, Android y web. Contiene diversas funciones para que cualquier desarrollador pueda combinar y adaptar la plataforma a medida de sus necesidades.

Principales servicios de Firebase:

- ***Firebase Analyctics:*** es una solución gratuita de medición de apps que proporciona estadísticas sobre el uso de las apps y la participación de los usuarios.

- ***Firebase Cloud Messaging:*** es una solución de mensajería multiplataforma que te permite enviar mensajes de forma segura y gratuita.

- ***Firebase Auth:*** Firebase Auth es un servicio que puede autenticar los usuarios utilizando únicamente código del lado del cliente. Incluye la autenticación mediante proveedores de inicio de sesión como Facebook, GitHub, Twitter, Google, Yahoo y Microsoft; así como los métodos clásicos de inicio de sesión mediante correo electrónico y contraseña. Además, incluye un sistema de administración del usuario por el cual los desarrolladores pueden habilitar la autenticación de usuarios con email y contraseña que se almacenarán en Firebase.

- ***Firebase Realtime database:*** Almacena y sincroniza datos con nuestra base de datos NoSQL alojada en la nube. Los datos se sincronizan con todos los clientes en tiempo real y se mantienen disponibles cuando la app no tiene conexión.
- ***Firebase Storage:*** Cloud Storage se creó para programadores de apps que necesitan almacenar y publicar contenido generado por usuarios, como fotos o videos.

[12] El *cloud computing* consiste en la posibilidad de ofrecer servicios a través de Internet.

- ***Cloud Firestore:*** Usa nuestra base de datos NoSQL flexible, escalable y en la nube a fin de almacenar y sincronizar datos para la programación en el lado del cliente y del servidor.

- ***Cloud Functions:*** Es un framework sin servidores que permite ejecutar de forma automática el código en backend en respuesta a solicitudes HTTPS. El código JavaScript o TypeScript se almacena en la nube de Google y se ejecuta en un entorno administrado. De esta manera no se necesita administrar ni escalar los propios servidores.

3.1.3. El sistema Liquid Galaxy

En este proyecto, el sistema Liquid Galaxy será utilizado como una parte extra de la aplicación, ya que no todos los usuarios disponen de esta tecnología. En el caso de los usuarios que disponen del sistema Liquid Galaxy, la aplicación se podrá conectar al Liquid Galaxy y se podrá mostrar información de la aplicación a través de las pantallas del sistema.

Liquid Galaxy es un proyecto de código abierto creado por el empleado de Google Jason Holt en el año 2008. Este sistema permite visualizar Google Earth en unas pantallas panorámicas, creando una alta experiencia de usuario. [22]

Un sistema Liquid Galaxy es un clúster de ordenadores donde cada nodo gestiona una pantalla. Uno de los nodos, generalmente el que gestiona la pantalla central, tiene la función de master y se hace cargo de recibir toda la información y replicarla a la resta de nodos utilizando paso de mensajes por SSH[13], permitiendo que las pantallas se comporten de manera sincronizada.

En la siguiente imagen se puede observar un sistema Liquid Galaxy de 7 pantallas:

Imagen 9: Sistema Liquid Galaxy de 7 pantallas

[13] SSH o Secure Shell, es un protocolo de administración remota que le permite a los usuarios controlar y modificar sus servidores remotos a través de Internet a través de un mecanismo de autenticación. [23]

En nuestro proyecto, se van a utilizar diferentes funciones descritas a continuación:

Conexión con el Liquid Galaxy

La comunicación con el sistema Liquid Galaxy se realiza en un solo sentido, solamente se reciben datos de fuera, pero el sistema no devuelve nada a las aplicaciones exteriores.

El Liquid Galaxy esta siempre escuchando por el fichero kmls.txt. En este fichero se encuentra el contenido de un fichero kml o la URL que apunta a un fichero KML y en el momento que se detecta algún cambio en este fichero, el Liquid Galaxy lo descarga y muestra el contenido en las pantallas. El fichero kmls.txt se puede editar desde fuera utilizando el protocolo SSH. [23]

Crear un POI

Un POI es un Punto de Interés es una ubicación o un punto específico. Para crear un POI se utilizan los siguientes parámetros:

- ***Longitud (este u oeste):*** distancia angular en grados con respecto al primer meridiano. Los valores al oeste del meridiano oscilan entre –180 y 0 grados. Los valores al este del meridiano oscilan entre 0 y 180 grados. El valor predeterminado es 0. Coordenadas en grados, minutos, segundos, dirección:
    - Grados (–180 a 180)
    - Minutos (0 a 59)
    - Segundos (00.00 a 59.99)
    - Dirección: Este u Oeste (E, W) Formato decimal (convertido de grados, minutos y segundos): La longitud puede variar de –179.59.59.99 W a 179.59.59.99 E

- ***Latitud (norte o sur):*** distancia angular en grados con relación al ecuador. Los valores al sur del ecuador oscilan entre –90 y 0 grados. Los valores al norte del ecuador varían de 0 a 90 grados. El valor predeterminado es 0. Coordenadas en grados, minutos, segundos, dirección:
    - Grados (–90 a 90)
    - Minutos (0 a 59)
    - Segundos (00.00 a 59.99)
    - Dirección: Norte o Sur (N, S) Formato decimal (convertido de grados, minutos y segundos): La latitud puede variar de –89.59.59.99 S a 89.59.59.99 N

- ***Altitud:*** altura o distancia del punto de acceso desde la superficie de la tierra en metros. Si no se proporciona, el valor predeterminado es 0. Los valores varían de 0 a 99999.

- ***Inclinación:*** los valores oscilan entre 0 y 90 grados (no puede ser negativo). Un valor de inclinación de 0 grados indica que se ve directamente desde arriba del

punto de acceso. Un valor de inclinación de 90 grados indica visualización a lo largo del horizonte. Los valores oscilan entre 0 y 90. El ángulo de acimut predeterminado es 0.

- ***Alcance:*** distancia en metros desde el punto especificado por la longitud y latitud hasta el punto donde se ve el punto de acceso (la posición de Observación) (alcance de la cámara sobre el nivel del mar). Los valores varían de 0 a 999999.

- ***Modo de altitud:*** indica cómo se interpreta el especificado para el punto de observación.

    - Sujetado al suelo: ignora la especificación y coloca la posición de observación en el suelo. Este es el valor predeterminado.
    - Relativo al suelo: se interpreta como un valor en metros sobre el suelo.
    - Absoluto: se interpreta como un valor en metros sobre el nivel del mar.
    - Extendido al suelo: indica si el punto de acceso está conectado o no a un mástil.

Crear marcas de posición

Las marcas de posición permiten marcar una posición en la superficie de la Tierra con un icono definido. La marca de posición (Placemark) más sencilla incluye solo un elemento de punto (<Point>), que especifica la ubicación de la marca de posición. Puedes especificar un nombre y un icono personalizado para una marca de posición y, si quieres, le puedes añadir otros elementos geométricos. [24]

Para crear una marca de posición en el Liquid Galaxy, se genera un archivo de tipo .kml con los atributos especificados anteriormente y se envía al master.

Realizar viajes

Los viajes se crean mediante la colocación de elementos específicos, en orden, dentro de un archivo KML. El archivo KML puede contener cualquier otro elemento KML legal junto con el viaje. [24]

En nuestro caso, se realizan viajes a una ciudad determinada o a un usuario en seleccionado. Para esto utilizaremos la longitud, latitud, altitud y nombre.

HTML descriptivo en las marcas de posición

En una marca de posición se pueden añadir enlaces, tamaños de fuentes, estilos y colores, además de especificar alineaciones de texto y tablas. Para escribir código HTML estándar dentro de una etiqueta de descripción (<description>), se puede hacer dentro de una etiqueta CDATA. [24] En nuestro caso, se utiliza para mostrar información en la pantalla sobre el usuario seleccionado.

Limpiar los KMLs

Cuando enviamos un archivo de tipo .kml al Liquid Galaxy, por tal de que se muestre en el Google Earth, hace falta editar el archivo */var/www/html/kmls.txt* y añadir la ruta del mismo, utilizando la IP del master y el puerto 81. Un ejemplo de ruta utilizada en el archivo es la siguiente: *http://ip_master:81/ruta_al_archivo_kml*

Cuando queremos que la información que se muestra, desaparezca, hace falta borrar las rutas añadidas en el fichero mencionado anteriormente.

3.1.4. Google Earth

Tal como se ha comentado anteriormente, Liquid Galaxy ejecuta el programa Google Earth. [25]

Google Earth es un programa informático que genera una representación 3D de la Tierra basada en imágenes de satélite. El programa mapea la Tierra mediante la superposición de imágenes obtenidas de fotografías satelitales y aéreas, y datos geográficos en un globo 3D. Google Earth es compatible con la gestión de datos geoespaciales tridimensionales mediante ficheros Keyhole Markup Language (KML), que se explicarán a continuación.

En la siguiente foto se puede observar la pantalla principal de Google Earth:

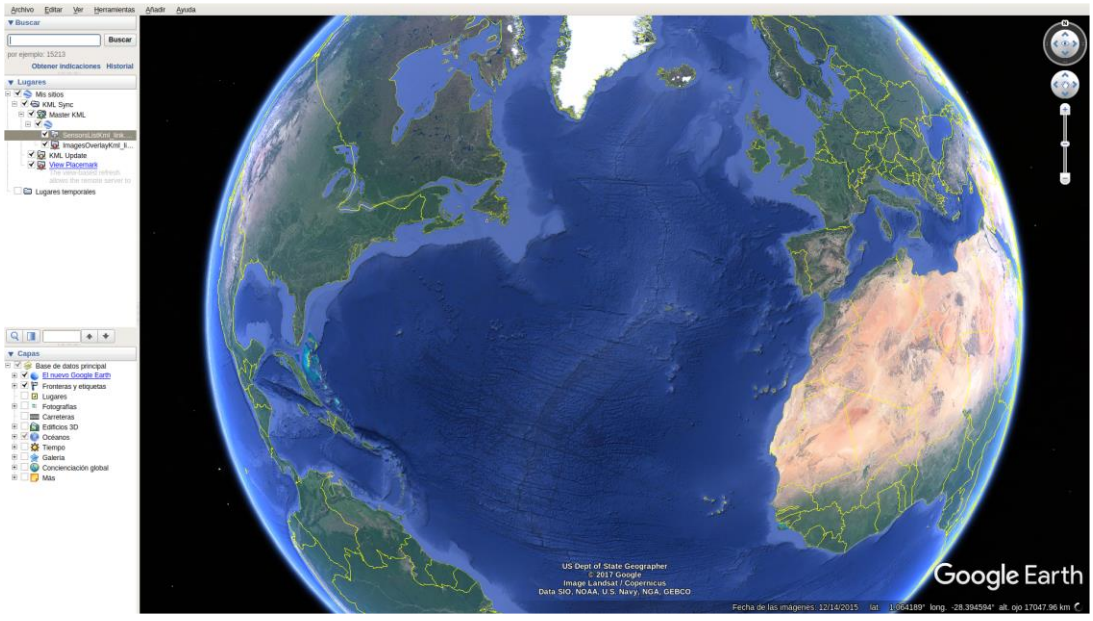


Imagen 10: Pantalla principal de Google Earth

3.1.5. KML

Keyhole Markup Language (KML) es un standard basado en el lenguaje de marcas en XML, que permite representar datos geográficos que especifican una característica, como por ejemplo una posición determinada. KML ha sido creado para ser utilizado con el Google Earth, el cual ha sido el primer programa capaz de ver y editar gráficamente ficheros KML. Un fichero KML incluye especificaciones para varias funciones para mostrar en Google Earth o Maps y otros programas tridimensionales de Earth o geobrowser. [24]

El conjunto de características de KML incluye marcas de posición, modelos 3D, descripciones de texto, imágenes, polígonos, etc. Cada ubicación tiene una longitud y latitud asociadas y se pueden proporcionar datos específicos de la vista, como el rumbo, la altitud y la inclinación, para definir la llamada "vista de cámara" para los datos geoespaciales. KML comparte parte de su gramática con el lenguaje de marcado de geografía, o GML, un lenguaje de marcado de Open XML definido para expresar datos y características geográficas. Los documentos KML a menudo se distribuyen en forma de archivos KMZ, que no son más que un documento KML comprimido dentro de un archivo con extensión kmz. Un archivo KMZ generalmente contiene un solo documento KML, invariablemente llamado "doc.kml" junto con imágenes de superposiciones e iconos a los que puede hacer referencia internamente.

A continuación, un ejemplo de fichero KML:

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
   <Style id="randomLabelColor">
      <LabelStyle>
         <color>ff0000cc</color>
         <colorMode>random</colorMode>
         <scale>1.5</scale>
      </LabelStyle>
   </Style>
   <Placemark>
      <name>LabelStyle.kml</name>
      <styleUrl>#randomLabelColor</styleUrl>
      <Point>
         <coordinates>-122.367375,37.829192,0</coordinates>
      </Point>
   </Placemark>
</Document>
</kml>
```

Imagen 11: Ejemplo fichero KML

El ejemplo de la Imagen 9 muestra solo la etiqueta Placemark con el punto de coordenadas (longitud, latitud, altitud). El estilo define colores, iconos, tamaños, etc. y es necesario

definir con <styleUrl> cuando se crea la marca de posición. Hay otras características, geometrías, vistas abstractas, etc.

Todos los elementos KML permiten crear archivos con una gran variedad de opciones que una vez lanzadas en Google Earth en el Liquid Galaxy generan una visualización y experiencia espectaculares para el usuario.

3.1.6. Sistema de control de versiones

Para realizar este proyecto se ha utilizado GitHub, la plataforma de desarrollo colaborativo para alojar proyectos utilizando el control de versiones Git. [26]

Todo esto es necesario para llevar un registro de los cambios en el código fuente de la aplicación. GitHub ha servido para guardar en la nube el proyecto y llevar un control de las diferentes versiones y etapas de desarrollo. También ha sido de gran ayuda para evitar y corregir posibles problemas debido a su retro compatibilidad.

Todo el código del proyecto puede ser encontrado en un repositorio en el GitHub. [64]

3.1.7. Google Play Services

Google Play Services es una librería creada por Google por la cual los desarrolladores pueden aprovechar las API[14] de Google fácilmente. [27]

Como en el caso de nuestra aplicación se han utilizado funciones como la ubicación, los teléfonos donde se instala la aplicación tendrán que contar con los Servicios de Google Play.

3.1.8. Librerías externas

Teniendo en cuenta que en Android Studio también utilizamos el lenguaje de programación Java para el desarrollo de la aplicación, se han utilizado varias librerías externas.

Una librería externa es un conjunto de clases, que poseen una serie de métodos y atributos. Lo realmente interesante de estas librerías para Java es que facilitan muchas operaciones. De una forma más completa, las librerías en Java nos permiten reutilizar código, es decir que podemos hacer uso de los métodos, clases y atributos que componen la librería evitando así tener que implementar nosotros mismos esas funcionalidades.

A continuación, se describirán brevemente las librerías utilizadas en el proyecto:

[14] Una API contiene un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones. API significa interfaz de programación de aplicaciones.

- ***Braintree Android card-form:*** es un formulario para usar con tarjetas de crédito o débito. [28]
- ***Currency EditText:*** es una implementación personalizada de EditText que permite formatear entradas numéricas basadas en monedas. [29]

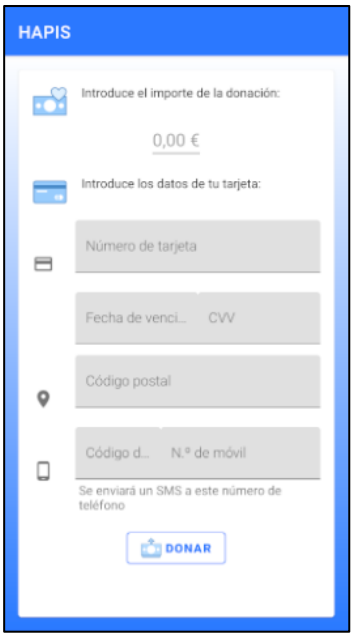


Imagen 12: Uso de la librería Braintree Card Form y Currency EditText

- ***Clans/Floating Action Button:*** utilizada para crear un menú flotante personalizado. [30]

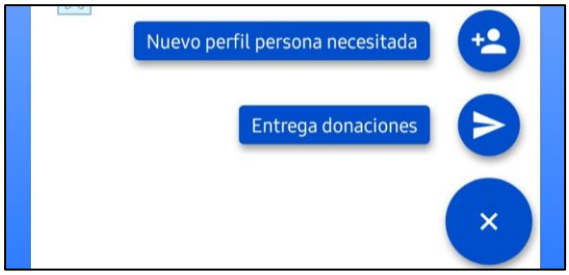


Imagen 13: Uso de la librería Floating Action Button

- ***Signature Pad:*** utilizada para integrar firmas en la aplicación. [31]

Imagen 14: Uso de la librería Signature Pad

- ***Glide:*** se ha utilizado para cargar las fotos de Firebase en el RecyclerView[15]. [32]

- ***YouTube Android Player API:*** se ha utilizado para reproducir los videos de Youtube en la aplicación. [33]

Imagen 15: Uso de la librería YouTube Android Player API

- ***JScH (Java Secure Channel):*** se ha utilizado para realizar la conexión SSH entre la aplicación móvil y el sistema Liquid Galaxy. [34]

- ***Android Image Cropper:*** se ha utilizado para poder cortar y rotar las imágenes de las personas necesitadas al subirlas. También se ha utilizado en la parte del Liquid Galaxy para cambiar la foto de una ciudad. [35]

Imagen 16: Uso de la librería Android Image Cropper

[15] Un RecyclerView es una versión más avanzada y flexible de las listas. Es un contenedor para mostrar grandes conjuntos de datos que se pueden desplazar de manera muy eficiente manteniendo un número limitado de vistas. [52]

3.2. Entorno de ejecución

El desarrollo de la aplicación en Android se realizará utilizando el lenguaje de programación Java, que se explicará brevemente a continuación. Teniendo en cuenta que cada vez más desarrolladores han sustituido el Java por otro lenguaje de programación, llamado Kotlin también se explicará, aunque no se ha utilizado en el desarrollo de este proyecto.

3.2.1. Java

Java es un lenguaje de programación, en este caso utilizado en el Android Studio, de propósito general orientado a objetos desarrollado por Sun Microsystems. También se puede decir que Java es una tecnología que no sólo se reduce al lenguaje, sino que además provee de una máquina virtual Java que permite ejecutar código compilado Java, sea cual sea la plataforma que exista por debajo; plataforma tanto hardware, como software (el sistema operativo que soporte ese hardware). El apoyo a esta tecnología viene dado por la gran cantidad de fabricantes que apoyan esta especificación de máquina virtual. [36]

Sun Microsystems describe al lenguaje Java de la siguiente manera:

- Simple
- Orientado a Objetos
- Tipado estáticamente
- Distribuido
- Interpretado
- Robusto
- Seguro
- de Arquitectura Neutral
- Multihilo
- con Recolector de basura (Garbage Collector)
- Portable
- de Alto Rendimiento: sobre todo con la aparición de hardware especializado y mejor software
- Dinámico

3.2.2. Kotlin

Aunque no se utiliza en el desarrollo de esta aplicación, he considerado importante destacar otro lenguaje de programación utilizado con Android Studio, Kotlin que es un lenguaje de programación principalmente orientado a objetos de calidad industrial y que se desarrolló por la empresa checa Jetbrains a partir de 2010.

Este lenguaje formal tiene un tipado estático y puede ser utilizado en servidores, en sitios web y en el sistema operativo de Apple (iOS), siendo uno de los oficiales para desarrollar aplicaciones Android, como lo reconoció Google en el año 2017.

Kotlin se ejecuta en la Máquina Virtual de Java (JMV) y es interoperable con Javascript y frente a otros lenguajes de programación reduce la repetición de código, lo que a su vez ahorra recursos y tiempo, facilitando la localización de errores en caso de que se produzcan.

Además de la reducción de líneas de código, que se estima aproximadamente en un 40 % con respecto a otros lenguajes, Kotlin gestiona los nulos de forma segura, de tal forma que no se van a producir los Null Pointer Exception (NPE), aunque si se necesita esta característica de nulabilidad, bastará con añadir "?" tras el nombre del tipo.[37]

3.3. Herramientas de desarrollo

A la hora de escoger las herramientas de desarrollo, se ha mirado sobretodo que sean de código abierto. De este modo, tanto el código como las herramientas pueden ser utilizadas sin ningún coste y además si alguien quiere aprovechar algo en un futuro podrá hacerlo sin ningún problema.

En este apartado se explicará brevemente el principal IDE (Entorno Integrado de Desarrollo) disponible para desarrollar aplicaciones en Android, que es el Android Studio. [38]

El Android Studio es el IDE oficial para el desarrollo de aplicaciones Android. Está basado en Intellij IDEA desarrollado por JetBrains. Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece incluso más funciones que aumentan la productividad cuando se desarrollan apps para Android, como las siguientes:

- Un sistema de compilación flexible basado en Gradle
- Un emulador rápido y cargado de funciones
- Un entorno unificado donde puedes desarrollar para todos los dispositivos Android

- Aplicación de cambios para insertar cambios de códigos y recursos a la aplicación en ejecución sin reiniciar la aplicación
- Integración con sistemas de gestión de proyectos y control de versiones de código y plantillas de código para ayudarte a compilar funciones de apps comunes y también importar código de ejemplo
- Variedad de marcos de trabajo y herramientas de prueba
- Herramientas de Lint para identificar problemas de rendimiento, usabilidad y compatibilidad de la versión, entre otros
- Compatibilidad con C++ y NDK
- Compatibilidad integrada con Firebase.

Cada proyecto de Android Studio incluye uno o más módulos con archivos de código fuente y archivos de recursos. Entre los tipos de módulos se incluyen los siguientes:

- Módulos de apps para Android
- Módulos de biblioteca

De manera predeterminada, Android Studio muestra los archivos de un proyecto en la vista de proyecto de Android, como se ve en la *Imagen 14*. Esta vista está organizada en módulos para que se pueda acceder rápidamente a los archivos fuente clave del proyecto.

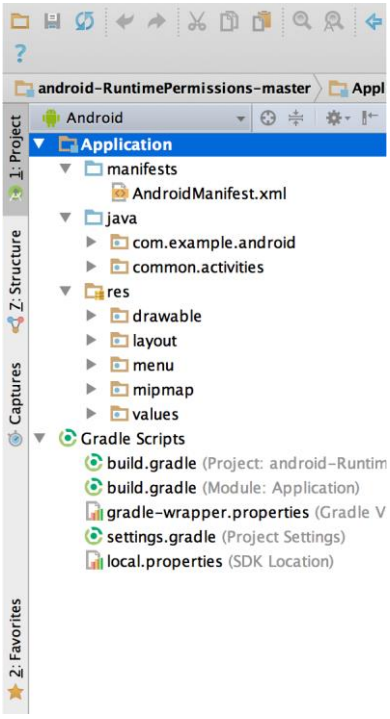


Imagen 17: Los archivos de proyecto en la vista de Android

Se pueden ver todos los archivos de compilación en el nivel superior de Secuencias de comando de Gradle y cada módulo de app contiene las siguientes carpetas:

- *manifests*: Contiene el archivo AndroidManifest.xml.
- *java*: Contiene los archivos de código fuente Java, incluido el código de prueba de JUnit.
- *res*: Contiene todos los recursos sin código, como diseños XML, strings de IU e imágenes de mapa de bits.

Android Studio usa Gradle como base del sistema de compilación, y el complemento de Android para Gradle proporciona capacidades específicas de Android. Este sistema de compilación se ejecuta en una herramienta integrada desde el menú de Android Studio, y lo hace independientemente de la línea de comandos.

4. Descripción de la funcionalidad y análisis

En este apartado se definen los requerimientos y se muestra una vista global de la arquitectura pensada para el sistema. Más adelante se tendrá en cuenta este catálogo de requisitos como base para el diseño de todos los aspectos de la aplicación. Es por ello por lo que la fase de análisis es de suma importancia para el devenir de todo proyecto software. Es en este capítulo donde se deben asentar las bases, a modo de cimientos, del proyecto, y, a partir de las cuales se construirá todo lo demás.

El proceso a seguir se basa en primer lugar en una buena definición de requisitos y en elegir una metodología de desarrollo acorde con el proyecto.

La arquitectura de la aplicación tendrá el formato de cliente- servidor, constituyendo el servidor la base de datos, y API's de acceso, y cliente, la aplicación móvil.

El catálogo de requisitos es la especificación del comportamiento que se espera de cualquier proyecto software. Como se ha visto en los capítulos anteriores, estudiando otras aplicaciones similares e incluyendo ideas originales, se ha predefinido una serie de requisitos que se consideran indispensables para el proyecto. A continuación, se muestra una enumeración y breve descripción de los requisitos establecidos para el diseño y desarrollo de la aplicación.

A continuación, se desglosan la funcionalidad y las características a modo de catálogo de requisitos, teniendo en cuenta tanto los requisitos funcionales como no funcionales.

4.1. Requerimientos funcionales

Los requerimientos funcionales de un sistema, son aquellos que describen cualquier actividad que este deba realizar, en otras palabras, el comportamiento o función particular de un sistema o software cuando se cumplen ciertas condiciones.

A continuación, se presentará un listado de requerimientos funcionales para el proyecto:

R.1. Requerimientos funcionales relacionados con el registro en la aplicación

R.1.1. Permitir registrar un perfil de usuario del tipo *Donante* o *Voluntario* a través de correo y contraseña.

R.2. Requerimientos funcionales relacionados con los usuarios registrados

R.2.1. Permitir editar un perfil de usuario registrado en la aplicación, una vez el usuario a iniciado sesión.

R.2.2. Permitir a los usuarios donar dinero para apoyar el desarrollo de la aplicación a través de un sistema de pago seguro o donar una mínima cantidad mirando un video publicitario patrocinado.

R.2.3. Permitir que el usuario se ponga en contacto con el encargado de gestionar todo a través de un formulario de contacto.

R.2.4. Permitir al usuario consultar una página de ayuda para poder entender mejor el funcionamiento de la aplicación.

R.2.5. Permitir al usuario consultar una página con información sobre la aplicación.

R.2.6. Permitir al usuario consultar en cualquier momento las condiciones de uso de la aplicación.

R.3. Requerimientos funcionales relacionados con el perfil de usuarios registrados como donantes

R.3.1. Mostrar al usuario un listado de perfiles de personas necesitadas.

R.3.2. Permitir la revisión de cada perfil, mostrando información sobre las experiencias de vida de la persona necesitada, la ubicación, el horario en el que la persona estará en dicha ubicación, las necesidades que tiene actualmente y una foto de perfil.

R.3.3. Dar la opción al usuario de escoger el tipo de ayuda que más le convence. El sistema mostrará una lista con las siguientes opciones: Comida, Ropa, Alojamiento, Trabajo y Productos de higiene personal.

R.3.4. Dar al usuario la opción de escoger si quiere ofrecer los recursos personalmente o entregarlos a través de un voluntario.

R.3.5. En el caso que el usuario quiere entregar los recursos personalmente tendrá acceso a un mapa con la ubicación temporal de la persona necesitada por tal de poder ir en el plazo de tiempo especificado y hacer la entrega personalmente.

R.3.6. En el caso que el usuario quiere entregar los recursos a través de un voluntario, el sistema permitirá al usuario elegir una ubicación, el día y la hora para encontrarse al voluntario y entregarle los recursos.

R.3.7. Los usuarios tendrán la opción de visualizar las personas necesitadas también a través de un mapa con marcadores o de una lista de mapas que permitirán mostrar el trayecto hasta llegar en la ubicación especificada.

R.4. Requerimientos funcionales relacionados con los usuarios que quieren ofrecer recursos personalmente

R.4.1. Los usuarios que entregarán los recursos personalmente tendrán la opción de compartir fotos a través de las redes sociales de la aplicación siempre y cuando se cumpla la política de privacidad, solicitando permiso a la persona necesitada para salir en la foto o sacar la foto de tal manera que la persona necesitada queda como anónima.

R.5. Requerimientos funcionales relacionados con el perfil de usuarios registrados como voluntarios

R.5.1. Permitir al usuario crear y completar el perfil de una persona necesitada.

R.5.2. Permitir al usuario editar y gestionar varios perfiles de personas sin recursos.

R.5.3. Permitir al usuario notificar a través de la aplicación cuando se ha puesto en contacto con un donante o cuando ha realizado la entrega de una donación.
.

R.5.4. Permitir a las personas necesitadas leer los términos y condiciones relacionados con la información personal que será publicada confirmando su consentimiento a través de una firma, antes de que el voluntario cree el perfil.

R.6. Requerimientos funcionales relacionados con el sistema Liquid Galaxy

R.6.1. El sistema tendrá una parte dedicada al sistema Liquid Galaxy.

R.6.2. Permitir al usuario que dispone de un sistema Liquid Galaxy, editar la configuración necesaria para la conexión de la aplicación con el sistema de pantallas Liquid Galaxy.

R.6.3. Permitir al usuario consultar una página de ayuda para ver cómo funciona el sistema.

R.6.4. Permitir al usuario realizar simples acciones con el sistema Liquid Galaxy como por ejemplo apagarlo o reiniciarlo, facilitando así el uso de dicho sistema.

R.6.5. Permitir al usuario consultar la lista de ciudades donde se encuentran registrados perfile de personas necesitadas.

R.6.6. Permitir al usuario ver una lista de usuarios existentes en una determinada ciudad.

R.6.7. Permitir al usuario seleccionar un perfil de una lista de usuarios y poder ver la ubicación y una ventana con información en el sistema Liquid Galaxy.

R.6.8. Permitir al usuario seleccionar en la aplicación diferentes estadísticas que podrán ser vistas en el sistema Liquid Galaxy.

4.2. Requerimientos no funcionales

Los requerimientos no funcionales representan características generales y restricciones de la aplicación o sistema que se está desarrollando.

A continuación, se presentará un listado de requerimientos no funcionales para la aplicación:

1. Producto

1.1. Concurrencia

1.1.1. La aplicación móvil será multiusuario, permitiendo la compartición de información entre los diferentes usuarios de la aplicación.

1.1. Portabilidad

1.2.1. La aplicación se adaptará a cualquier dispositivo Android 6.0 o superior (API 23). *Smartphones* o *tablets.*

1.3. Usabilidad

1.3.1. Interfaz simple e intuitiva, proporcionando un alto nivel de interactividad y usabilidad.

1.3.2. La aplicación dispondrá de idioma castellano e inglés.

1.4. Hardware

1.4.1. Cámara disponible en los dispositivos de los voluntarios o los usuarios que quieren compartir fotos a través de las redes sociales de la aplicación.

2. Organización

2.1. Implementación

2.1.1. Plataforma Android utilizada para la app móvil. Se quiere llegar hasta la versión *Marshmallow* (versión mínima nivel de API: 23), ya que aún cubre un 15.1% de los dispositivos actuales.

3. Externos

3.1. Legislativo

3.1.1. Privacidad

3.1.1.1. La gestión de datos se ajustará a los requisitos de la Ley Orgánica de Protección de Datos, con el fin de preservar la privacidad en el tratamiento de los datos personales.

3.1.2. Seguridad

3.1.2.1. Para poder utilizar la aplicación hay que autentificarse.

3.1.2.2. Las personas que necesitan ayuda, tendrán la opción de no revelar su nombre real o publicar fotos en las que sale la cara, en caso que así lo desean.

3.1.2.3. El usuario será el único que podrá ver sus datos personales, las fotos que compartirá se harán a través de una cuenta de red social oficial de la aplicación.

4.3. Diagrama de casos de uso

A continuación, se mostrará el diagrama de casos de uso de la aplicación. El diagrama de casos de uso representa la forma en como un usuario o entidad externa (Actor) opera con el sistema en desarrollo, además de la forma, tipo y orden en como los elementos interactúan (operaciones o casos de uso). Un diagrama de casos de uso consta de los siguientes elementos: [39]

- Actor
- Casos de uso
- Relaciones de Uso, Herencia entre actores y Comunicación

En nuestro caso, se mostrarán dos diagramas de casos de uso:

- Diagrama de casos de uso sin la parte extra del Liquid Galaxy
- Diagrama de casos de uso de la parte del Liquid Galaxy

Para realizar los diagramas de casos de uso se ha utilizado la herramienta online diagramas.net que es una herramienta de creación y edición de diagramas libre que permite la integración con diversas plataformas. Además, permite trabajar en línea pudiendo guardar los diagramas en Google Drive, OneDrive o en local. [40]

4.3.1. Diagrama de casos de uso sin la parte extra del Liquid Galaxy

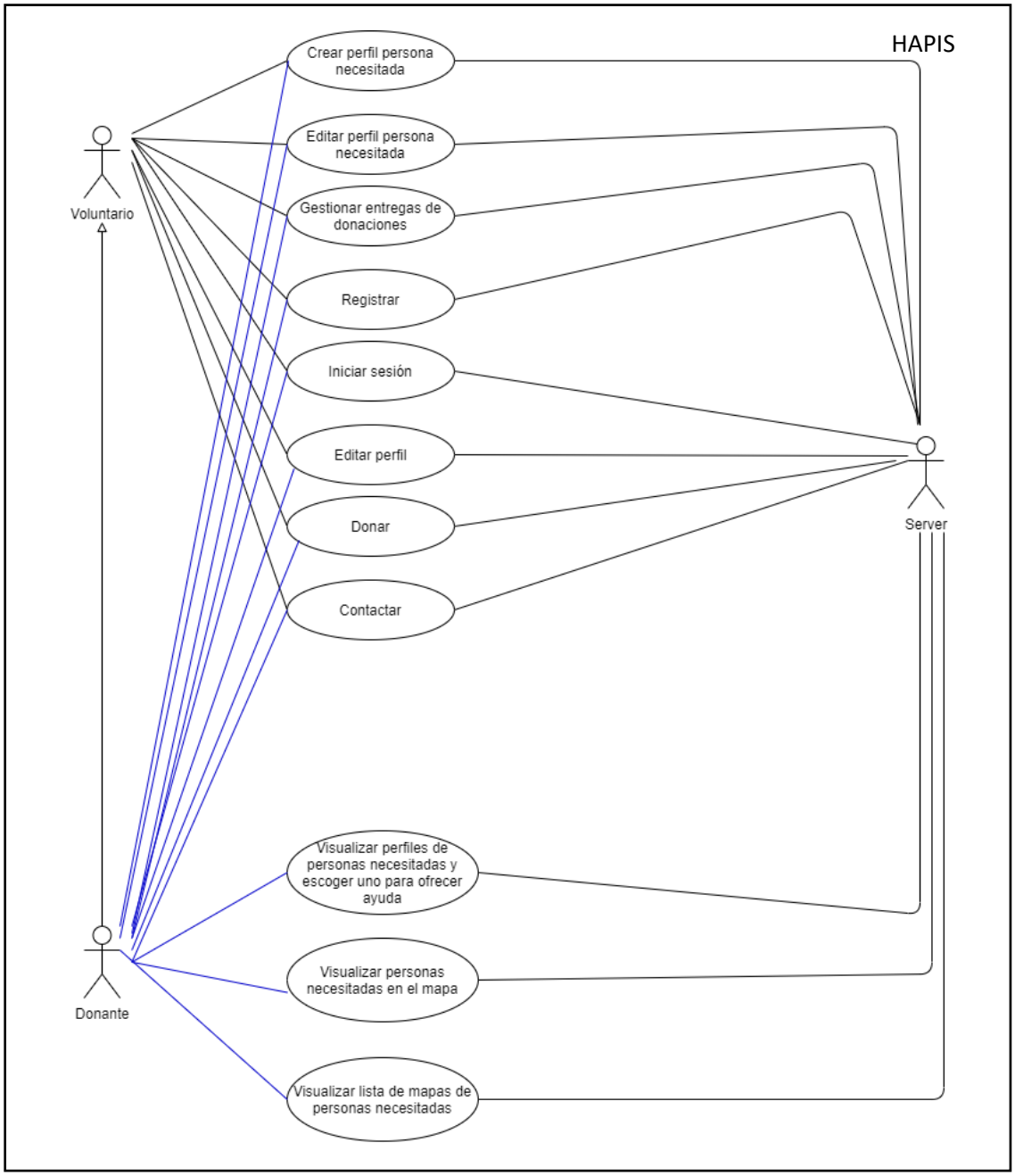


Imagen 18: Diagrama de casos de uso sin la parte extra del Liquid Galaxy

4.3.2. Diagrama de casos de uso de la parte del Liquid Galaxy

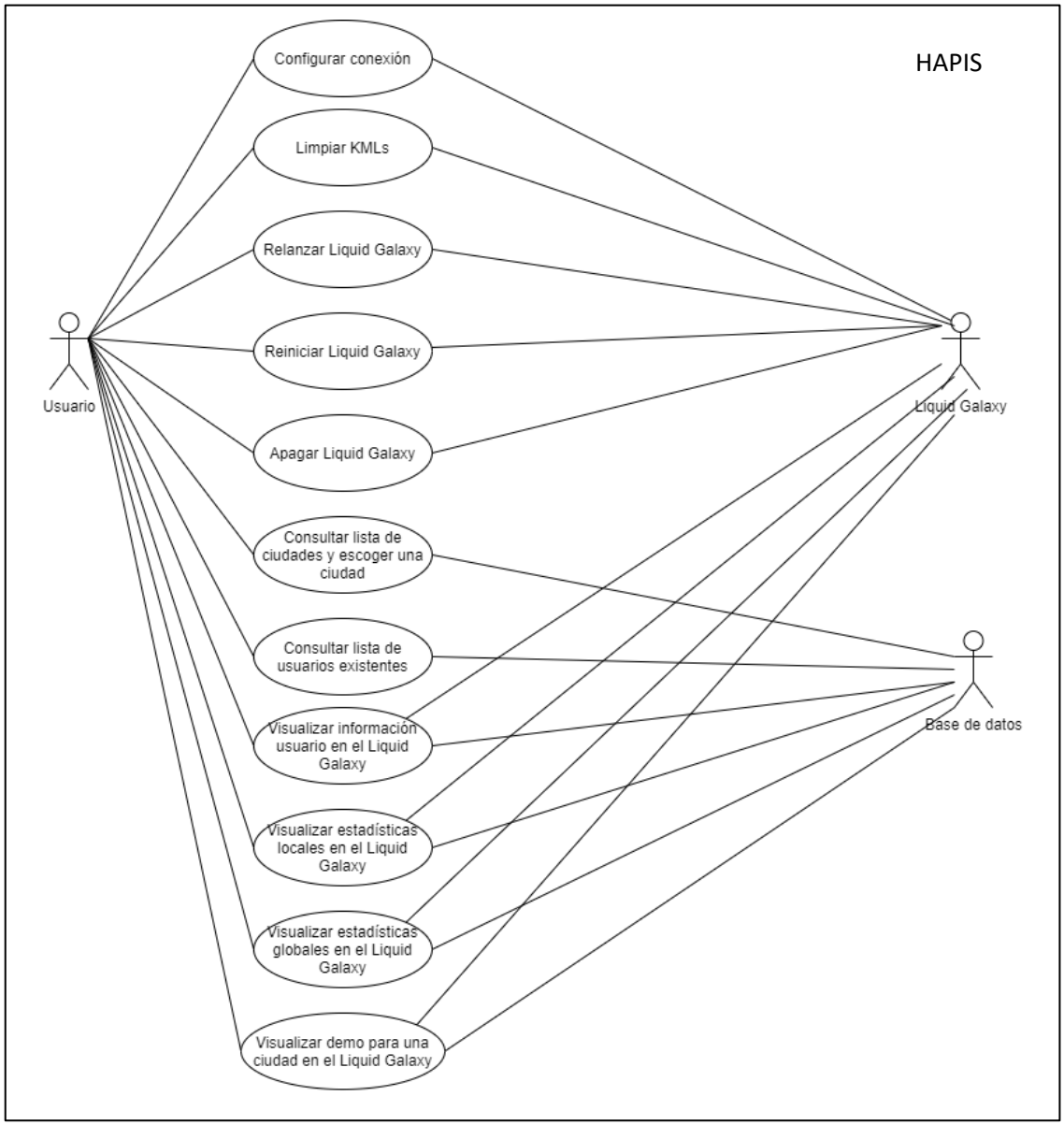


Imagen 19: Diagrama de casos de uso de la parte del Liquid Galaxy

4.4. Descripción de los casos de uso

Los casos de uso son descriptores analíticos de las interacciones que se producen entre usuarios o entidades externas (ex: Liquid Galaxy) y nuestra aplicación. No se trata de un documento técnico, sino que definimos en un lenguaje entendedor como utilizaremos la aplicación describiendo casos concretos que nos ayudaran a definir las interacciones en la aplicación. Antes de empezar, definiremos los actores que intervienen en el caso de uso:

- ***Usuario:*** persona que utilizará la aplicación. En nuestro caso puede ser de dos tipos: voluntario o donante.
- ***Backend:*** el servicio de bases de datos que utilizaremos para guardar y gestionar los datos de la aplicación. En este caso, Firebase.
- ***Liquid Galaxy:*** el sistema de pantallas panorámica que utiliza Google Earth y en el cual podemos visualizar nuestros datos.

A continuación, se describirán los tres casos de uso que se han considerado más importantes. El resto de casos de uso se podrán consular el Anexo I.

4.4.1. Caso de uso 1: Registrar usuario

Actores: usuario (donante o voluntario)

Precondición: el usuario que se crea no está registrado en el sistema

Postcondición: se creará un nuevo perfil de usuario con los datos introducidos

Actor	Sistema	Backend
1. El usuario quiere registrarse		
	2. El sistema muestra una pantalla con dos opciones a escoger: Registro como donante o Registro como voluntario	
3. El usuario escoge una de las dos opciones		
	4. El sistema muestra una pantalla con los datos a introducir para crear un nuevo usuario	
		5. El backend procesa los datos

	6. El sistema muestra en la pantalla un mensaje conforme el usuario se ha creado correctamente	
7. El usuario se ha registrado correctamente.		

Tabla 10: Caso de uso 1: Registrar usuario

Cursos alternativos:

3a – El usuario escoge la opción "Saber más":

El sistema muestra un mensaje con información sobre cada tipo de perfil

4a, 5a, 6a – No hay conexión a Internet:

El sistema muestra un mensaje en la pantalla

4b – El usuario ya existe:

El sistema devuelve un mensaje de error diciendo que el usuario ya existe.

4c – El usuario no acepta los términos de privacidad y las condiciones de uso:

El sistema devuelve un mensaje de error conforme el registro no se puede efectuar sin aceptar los términos y condiciones legales.

Requisititos especiales:

Para registrar un nuevo perfil de usuario hace falta una conexión a Internet.

4.4.2. Caso de uso 2: Crear perfil persona necesitada

Actores: usuario registrado como voluntario, persona necesitada

Precondición: el usuario ya está registrado y ha iniciado sesión en el sistema

Postcondición: se creará un nuevo perfil de persona necesitada con los datos introducidos

Actor	**Sistema**	**Backend**
1. El usuario quiere crear un nuevo perfil de persona necesitada		

	2. El sistema muestra una pantalla con la condiciones de privacidad necesarias para el registro	
3. El usuario entrega el teléfono móvil a la persona necesitada para leer las condiciones y realizar una firma si está de acuerdo con todo		
	4. El sistema muestra una pantalla para que la persona necesitada introduzca su nombre y apellidos	
		5. El backend procesa los datos
	6. El sistema muestra en la pantalla un formulario para que el voluntario introduzca los datos de perfil , de localización, y necesidades de la persona para la que crea el perfil	
		7. El backend procesa los datos
	8. El sistema muestra un mensaje conforme el perfil se ha creado con éxito	
7. El perfil de la persona necesitada se ha creado correctamente.		

Tabla 11: Caso de uso 2: Crear perfil persona necesitada

Cursos alternativos:

2a – La persona necesitada no está de acuerdo con las condiciones de privacidad:

El sistema cancela la creación del nuevo perfil.

4a, 5a, 6a – No hay conexión a Internet:

El sistema muestra un mensaje en la pantalla.

6b – El usuario ya existe:

El sistema devuelve un mensaje de error diciendo que el usuario ya existe.

6c – Alguno de los campos no cumplen con las condiciones programadas:

El sistema devuelve un mensaje de error diciendo que algo está mal.

Requisititos especiales:

Para crear un nuevo perfil de persona necesitada hace falta una conexión a Internet.

4.4.3. Caso de uso 3: Visualizar perfil de persona necesitada en el Liquid Galaxy

Actores: usuario

Precondición: el usuario dispone de un sistema Liquid Galaxy conectado a la aplicación

Actor	Sistema	Backend	Liquid Galaxy
1. El usuario quiere visualizar un perfil de persona necesitada de una ciudad determinada			
	2. El sistema muestra una pantalla en la que el usuario puede escoger una ciudad		
		3. El backend devuelve las ciudades en las que hay registradas personas necesitadas	
4. El usuario escoge una ciudad			
			5. El Liquid Galaxy realiza un viaje a la ciudad seleccionada
	6. El sistema muestra una pantalla en la que el usuario puede escoger entre Personas necesitada, Donantes o Voluntarios		
7. El usuario selecciona Personas necesitadas			
		8. El backend devuelve todas las	

		personas necesitadas registradas en la ciudad escogida anteriormente	
			9. El Liquid Galaxy muestra marcadores para todas las personas necesitadas existentes en la ciudad
	10. El sistema muestra en la pantalla una lista con todas las personas necesitadas existentes en la ciudad escogida, cada una con tres opciones: Bio, Transacciones u Orbita		
11. El usuario selecciona una persona necesitada de la lista			
		12. El backend devuelve la información de la persona seleccionada	
			13. El Liquid Galaxy muestra un cuadro con la información básica de la persona seleccionada

Tabla 12: Caso de uso 3: Visualizar perfil persona necesitada en el Liquid Galaxy

Cursos alternativos:

10a – El usuario selecciona el botón *Bio* para la persona necesitada:

El Liquid Galaxy muestra un cuadro adicional con información sobre la historia de vida de la persona necesitada.

10b – El usuario selecciona el botón *Transacciones* para la persona necesitada:

El Liquid Galaxy muestra un cuadro adicional con información sobre las transacciones realizadas para este usuario.

10c – El usuario selecciona el botón *Orbita* para la persona necesitada:

El Liquid Galaxy gira entorno a las coordenadas donde se encuentra la persona necesitada.

5a, 9a, 13a – La conexión con el Liquid Galaxy no se ha establecido correctamente:

Los datos no se mostrarán hasta que la conexión es correcta.

Requisititos especiales:

Para visualizar un perfil de persona necesitada en el Liquid Galaxy hace falta una conexión a Internet, un sistema Liquid Galaxy y la conexión funcionando entre la aplicación y el sistema Liquid Galaxy.

4.5. Especificaciones del sistema

Hay dos opciones de utilizar nuestra aplicación:

- ***Sin la parte extra del Liquid Galaxy:*** en este caso no se necesita ningún sistema especial, solamente un dispositivo móvil o Tablet. El dispositivo tiene que tener una conexión a Internet. Para una correcta visualización de los mapas hace falta que el dispositivo disponga de los servicios de Google Play.

- ***Con la parte extra del Liquid Galaxy:*** en este caso el usuario tiene que disponer, aparte del dispositivo móvil, de un sistema de mínimo 3 pantallas conectadas entre sí, con el sistema Liquid Galaxy Instalado. Los 3 ordenadores tendrán que tener instalado el sistema operativo Ubuntu 16.04.

4.5.1. Recursos hardware

Para la fase de desarrollo, principalmente se ha utilizado el siguiente equipamiento hardware:

- ***Teléfono móvil Samsung Galaxy A40:***
  - *Procesador:* Exynos 7904 (Ocho núcleos hasta 1.8 GHz)
  - *Memoria RAM:* 4GB
  - *Versión de Android:* 10 (Android Q)

- ***Tablet Acer B3-A40:***
  - *Procesador:* MTK MT 8167
  - *Memoria RAM:* 2GB
  - *Versión de Android:*7.0 (Android Q)

- ***3 ordenadores Gigabyte i7:***

    - *Procesador:* Intel Core i7
    - *Memoria RAM:* 4 GB
    - *Versión de Linux:* 16.04 64-bits

- ***3 monitores Benq GL2450:***
    - *Dimensiones:* 579 x 351 x 63 mm
    - *Pantalla:* 1920x1080 pixeles
    - *Frecuencia de actualización: 50 Hz – 76 Hz*

- ***Ordenador portátil Lenovo:***
    - *Procesador:* Intel Core i7 – 5500U
    - *Memoria RAM:* 8 GB
    - *Versión de Windows:* Windows 10 Home

4.5.2. Recursos software

Para la realización de este proyecto principalmente se ha utilizado el siguiente software:

- Android Studio 4.0.1 [41]
- Ubuntu 16.04 [42]
- Google Earth Pro 7.1.8 [43]

4.5.3. Restricciones técnicas

Algunas de las funcionalidades utilizadas en la aplicación solamente funcionan a partir de una cierta versión de Android. Como versión mínima de Android la API 23 (Marshmallow) y la aplicación funcionará correctamente en los dispositivos que tengan esta versión o una versión superior instalada.

En cuanto a la parte del Liquid Galaxy, el sistema necesita ordenadores con la versión de Ubuntu 16.04 instalada, sino el Google Earth no funcionará correctamente.

5. Diseño de la aplicación

5.1. Modelo relacional

El modelo relacional es un modelo de datos desarrollado inicialmente por E.F. Codd en el año 1969 y se ha establecido como el principal modelo de datos para aplicaciones comerciales de procesamiento de datos, permitiendo modelar problemas del mundo real. El modelo se basa en estructuras, llamadas tablas, basadas en conjuntos de relaciones y cada una de estas relaciones se denominan entidades. Cada entidad corresponde a una fila de la tabla. Cada columna corresponde a un atributo y cada atributo tiene un dominio asociado, que es conjunto de valores legales que puede tener el atributo. [44]

A partir de aquí, se puede determinar la identidad de cada una de las relaciones y la información que guarda, a través de un esquema compuesto por:

- Nombre de la relación
- Nombre de los atributos

En caso de la aplicación HAPIS, el esquema del modelo relacional está a continuación:

Usuario (Correo_electrónico, nombre_usuario, nombre, apellido, teléfono, dirección, ciudad, país, latitud, longitud, altitud)

Voluntario (*Correo_electrónico*, perfiles_creados)

Donante (*Correo_electrónico*, donaciones_personales, donaciones_voluntario)

Persona_necesitada (Nombre_usuario, fecha_nacimiento, teléfono, historia vida, ubicación, horario, necesidad, ciudad, país, latitud, longitud, imagen, donaciones_personales, donaciones_voluntario)

Formulario_contacto (*Correo_electrónico*, mensaje, asunto)

Donación (*Correo_electrónico*, tipo_donación, nombre_usuario_persona_necesitada)

A_través_de_voluntario (*Correo_electrónico*, teléfono_donante, fecha_encuentro, hora_encuentro, lugar_encuentro, nombre_usuario_donante, entregado, tipo_donación)

A continuación, se pueden observar las relaciones mencionadas anteriormente en un diagrama:

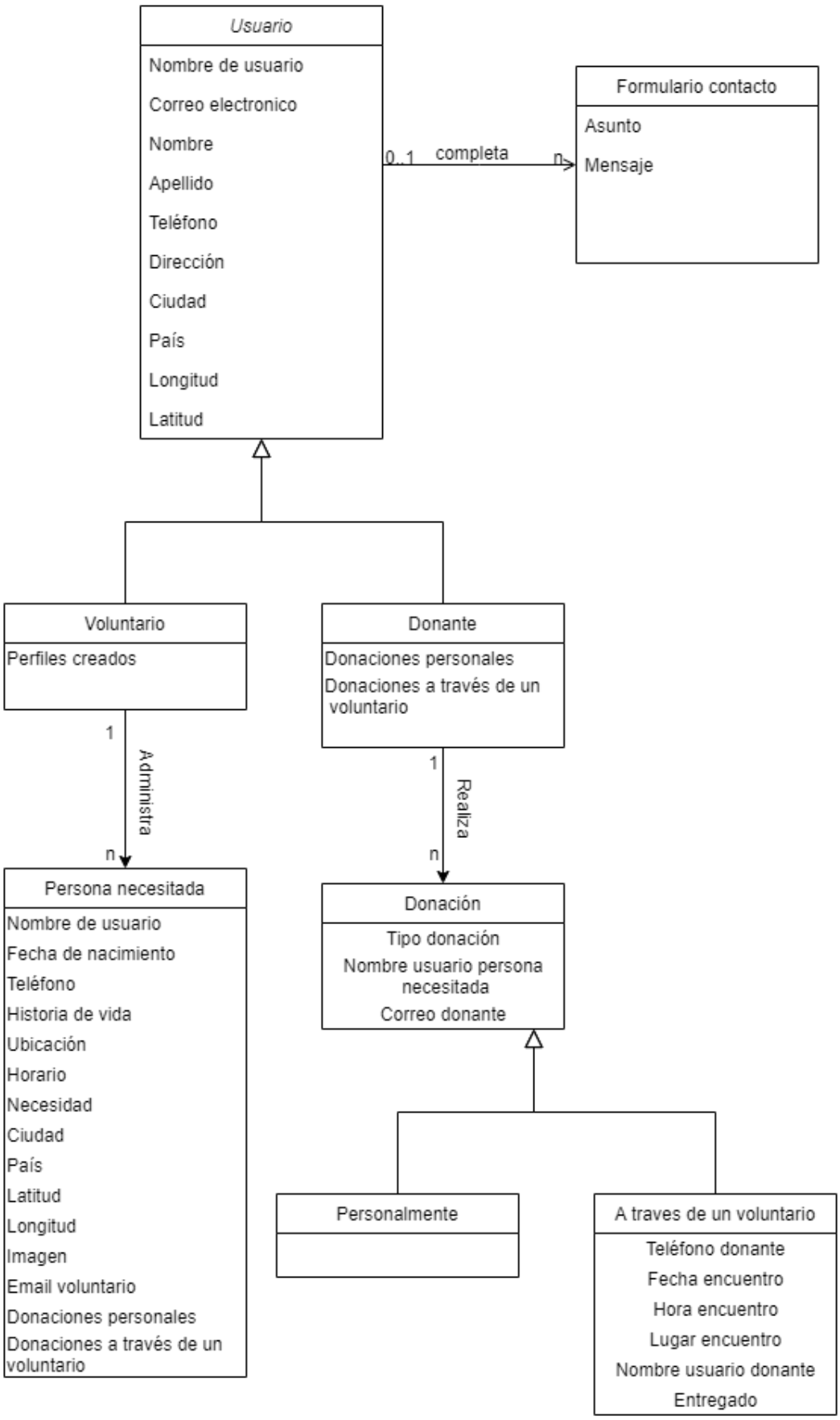


Imagen 20: Diagrama entidad – relación

5.2. Prototipado

En un principio, para tener una idea sobre la estructura del proyecto, se ha realizado un prototipo utilizando la herramienta Justinmind, que es una herramienta de creación de prototipos de aplicaciones web y móviles. Es una solución excelente para el prototipo de una aplicación, ya que tiene una interfaz muy fácil de utilizar y aparte dispone de varias plantillas para empezar el diseño. [45]

A continuación, se encuentran algunas pantallas del prototipo realizado con la herramienta Justinmind. Tal como se podrá observar, el diseño inicial es muy diferente del diseño final, ya que las ideas se han ido adaptando y las funcionalidades se han cambiado según las necesidades.

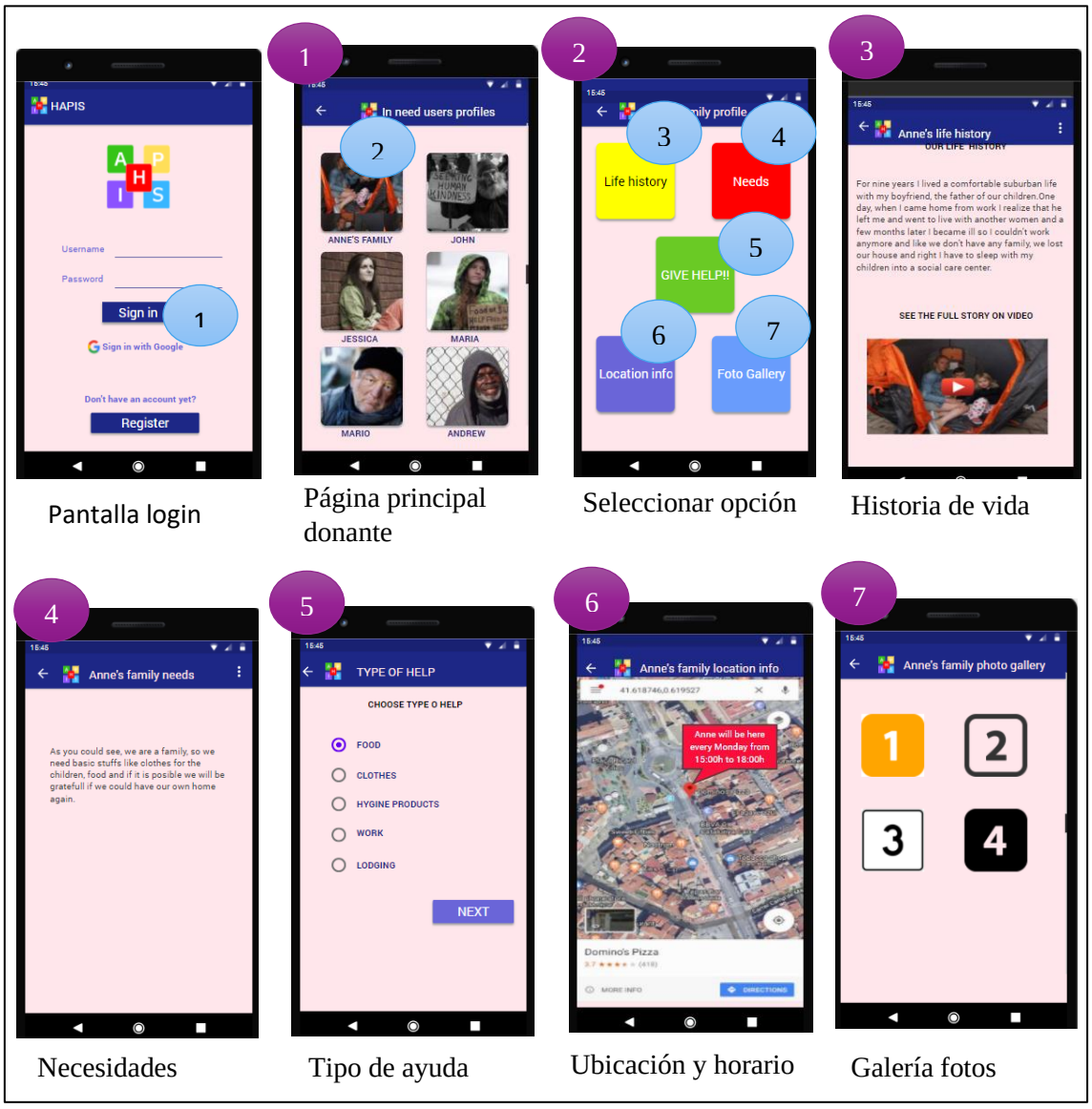


Imagen 21: Primer prototipo utilizando Justinmind

5.3. Diseño de la interfaz

En este caso, la aplicación está dividida en dos partes principales:

- Una parte en la cual el usuario se puede registrar y utilizar la aplicación con normalidad.
- Una parte que solamente funciona si el usuario tiene un sistema Liquid Galaxy conectado. Teniendo en cuenta que esta parte de la aplicación se utilizará en lugares públicos u eventos, se enseñará la interfaz de usuario diseñada para ser utilizada con un dispositivo de tipo Tablet.

A continuación, se mostrará un mapa de navegación y se explicará brevemente cada pantalla.

5.3.1. Mapa de navegación entre pantallas

La navegación entre pantallas, se hará en 4 partes principales:

- La navegación para un donante.
- La navegación para un voluntario.
- La navegación común para los dos tipos de usuario (que se encuentra en las opciones del menú lateral de la aplicación)
- La navegación para un usuario que dispone de Liquid Galaxy.

Para realizar los mapas de navegación se ha utilizado la herramienta Adobe XD [46] y del Google Drive, Dibujos de Google para realizar la conexión entre pantallas. [47]

5.3.1.1. Navegación para usuario de tipo donante

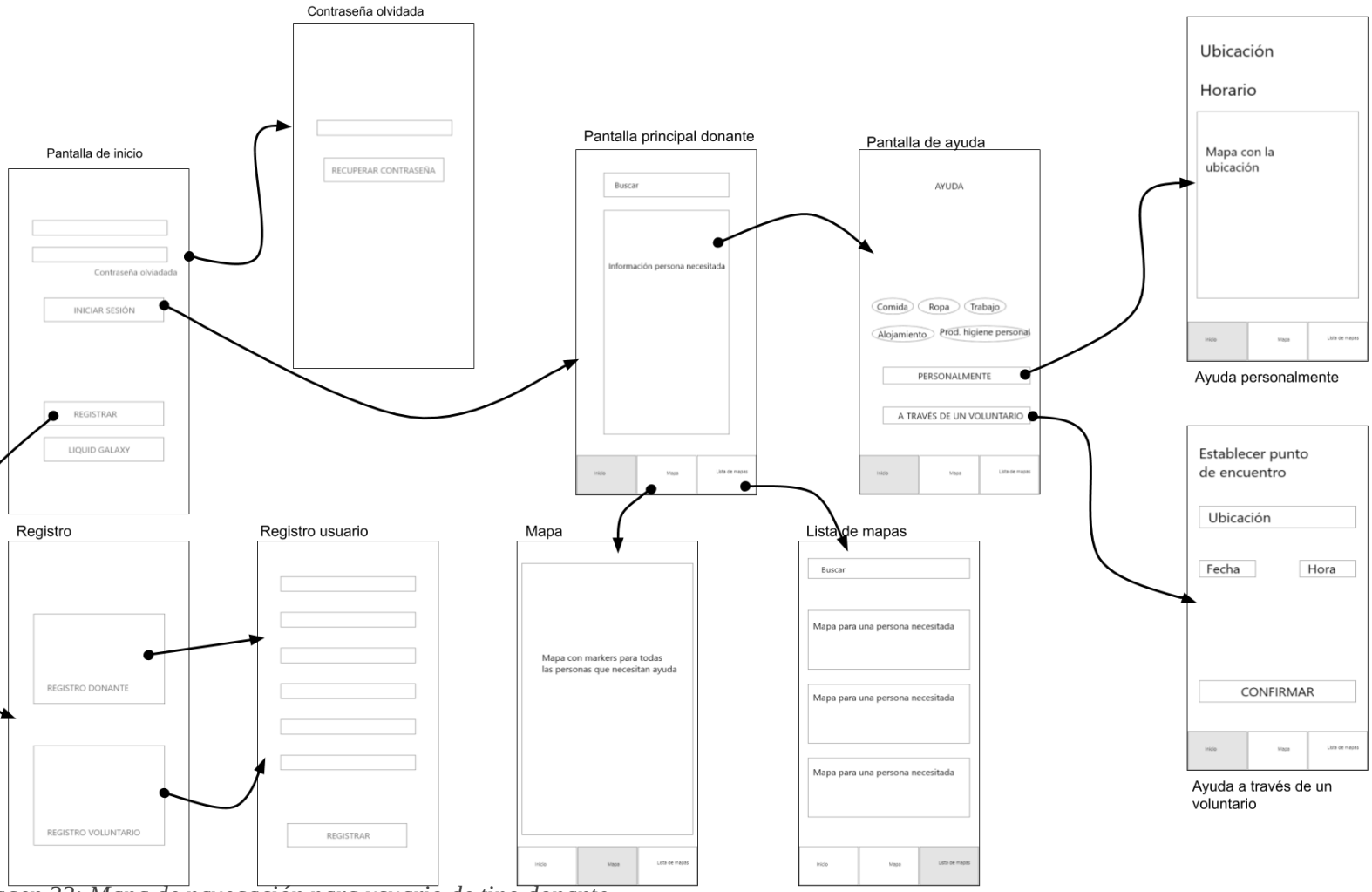


Imagen 22: Mapa de navegación para usuario de tipo donante

5.3.1.2. Navegación para usuario de tipo voluntario

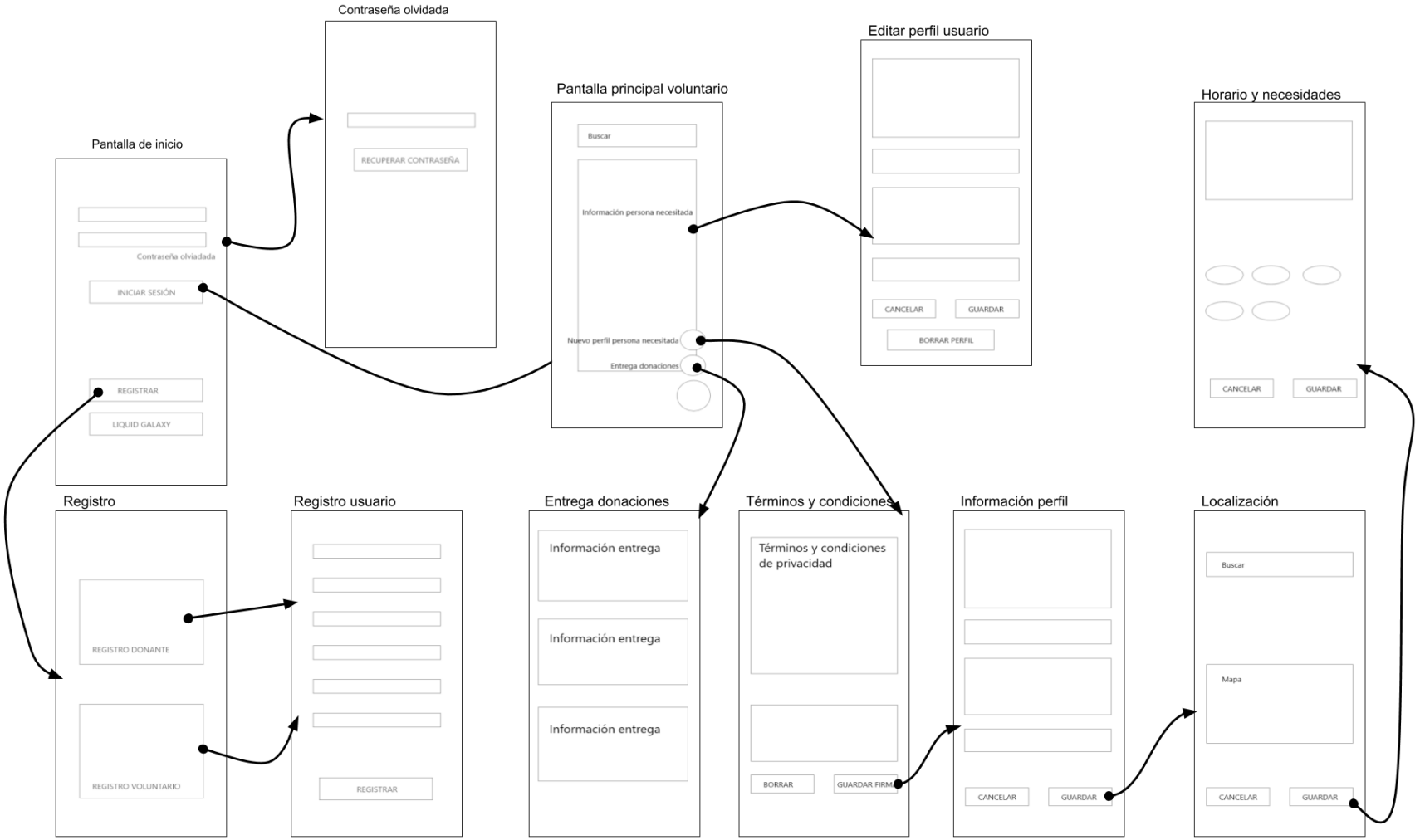


Imagen 23: Mapa de navegación para usuario de tipo voluntario

5.3.1.3. Navegación común para los dos tipos de usuario

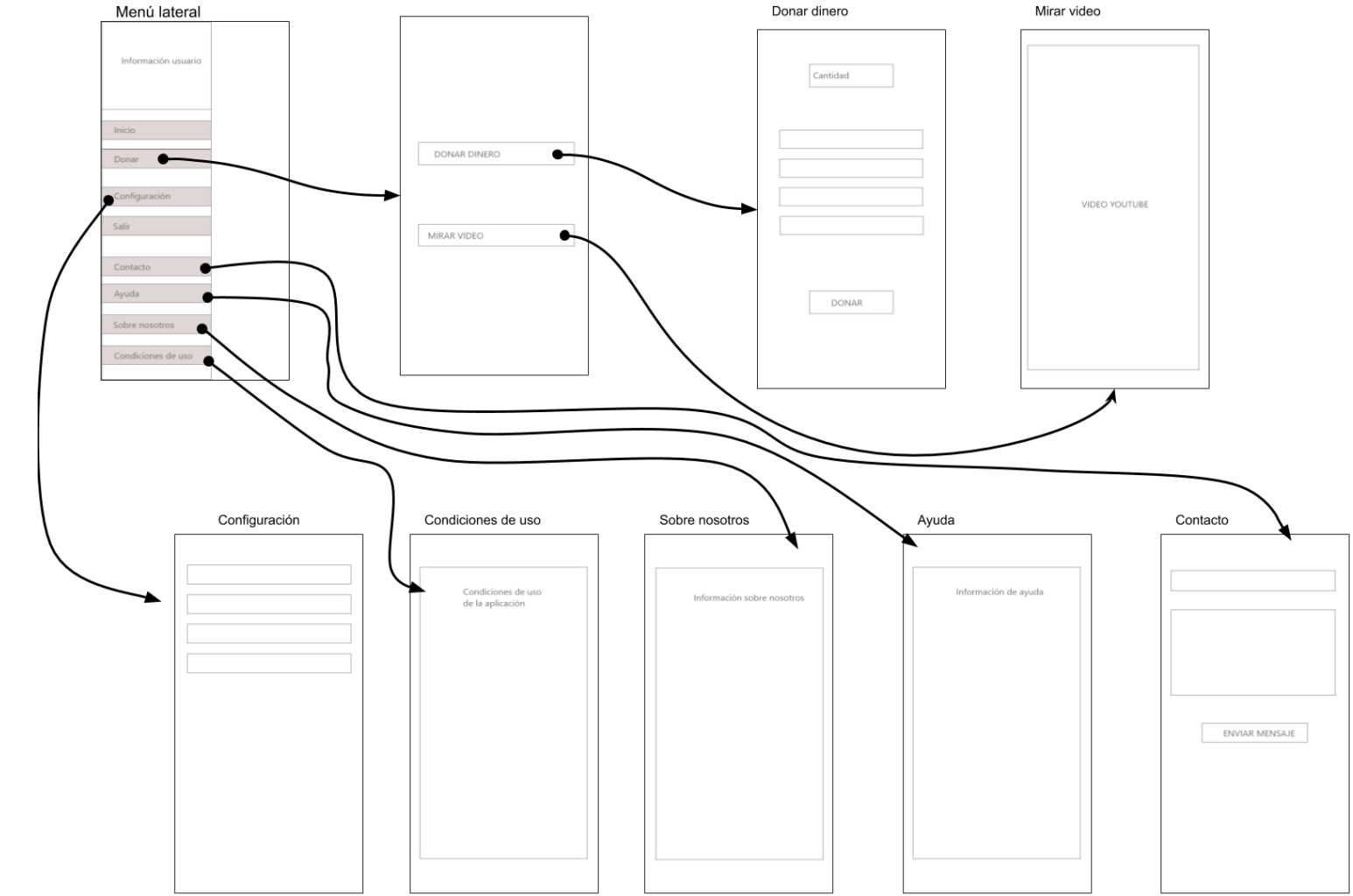


Imagen 24: Mapa de navegación común para los dos tipos de usuario

5.3.1.4. Navegación para la parte del Liquid Galaxy

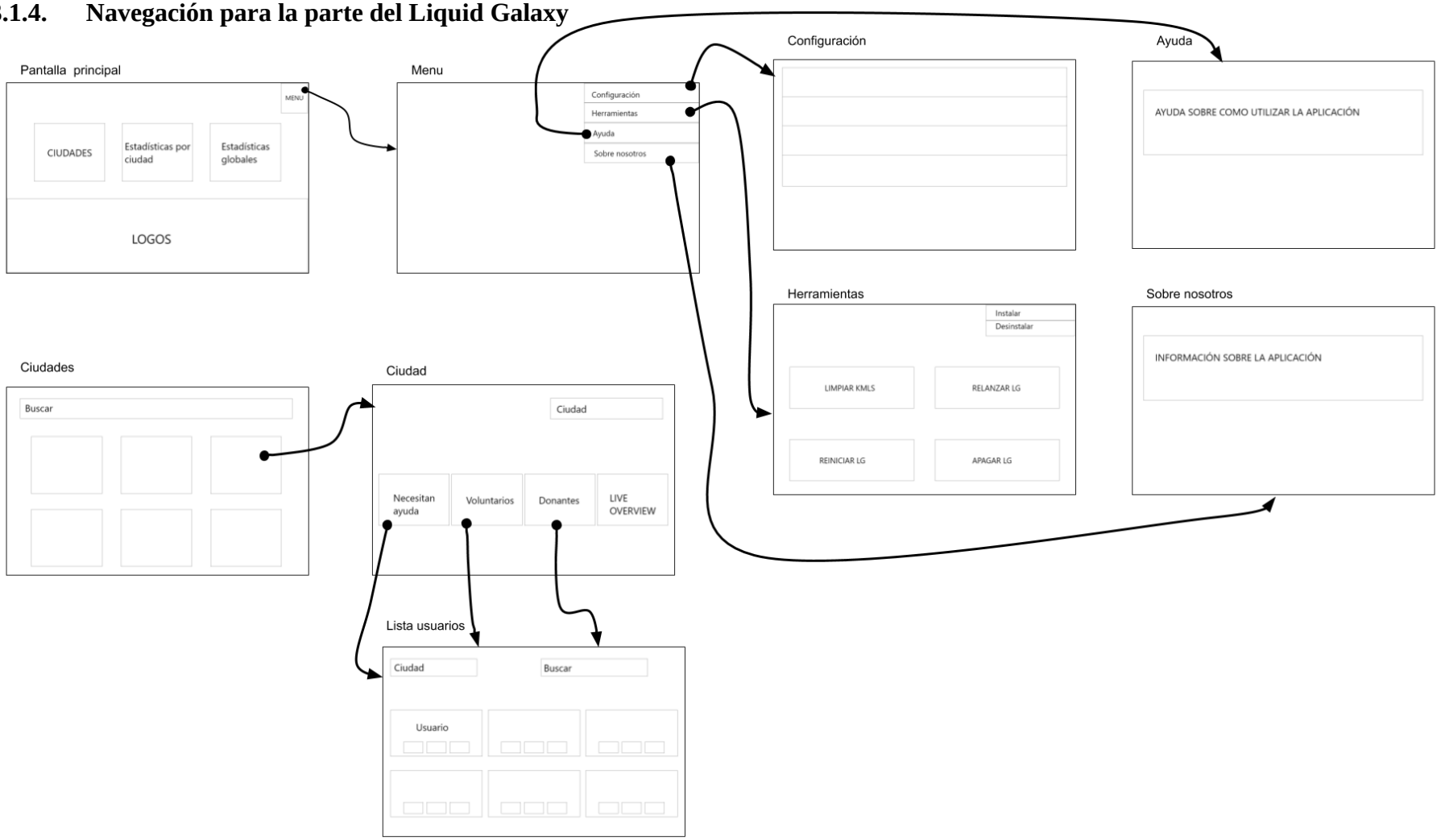


Imagen 25: Mapa de navegación parte del Liquid Galaxy

5.4. Estilo de navegación

El diseño de la aplicación se ha realizado de tal manera que la navegación sea lo más simple posible y fácil de usar para cualquier usuario.

Antes de explicar el diseño de la aplicación, se explicarán algunos términos que serán utilizados a lo largo de este capítulo y del capítulo de implementación:

- ***Actividad (Activity):*** Una actividad proporciona la ventana en la que la aplicación dibuja su Interfaz de Usuario. Por lo general, esta ventana llena la pantalla, pero puede ser más pequeña y flotar sobre otras ventanas. La mayoría de las aplicaciones contienen varias actividades, habiendo una pequeña dependencia entre ellas. [48]

- ***Fragmento:*** Un fragmento representa un comportamiento o una parte de la interfaz de usuario. Se pueden combinar varios fragmentos en una sola actividad para crear una Interfaz de Usuario multipanel y volver a usar un fragmento en diferentes actividades. Se puede pensar en un fragmento como una sección modular de una actividad que tiene un ciclo de vida propio, que recibe sus propios eventos de entrada y que puedes agregar o quitar mientras la actividad se esté ejecutando (algo así como una "subactividad" que puedes volver a usar en diferentes actividades). Un fragmento siempre debe estar alojado en una actividad y el ciclo de vida del fragmento se ve afectado directamente por el ciclo de vida de la actividad anfitriona. [49]

- ***Material Design:*** es una normativa de diseño enfocado en la visualización del sistema operativo Android, además en la web y en cualquier plataforma. [50]

- ***NavigationDrawer:*** es un elemento de interfaz definido por Material Design consistente en el típico menú lateral deslizante, generalmente desde la izquierda. Suele encontrase en la pantalla principal de la app y contar con un botón para desplegarlo, aunque también puede ser abierto con un gesto de desplazamiento. [51]

- ***RecyclerView:*** este widget es un contenedor para mostrar grandes conjuntos de datos que se pueden desplazar de manera muy eficiente manteniendo un número limitado de vistas. El widget RecyclerView se usa cuando se tienen colecciones de datos cuyos elementos cambian en tiempo de ejecución según la acción del usuario o los eventos de la red. [52]

- ***ViewPager:*** se utiliza para desplazarse entre los diferentes fragmentos. [53]

- ***TabLayout:*** Un objeto TabLayout proporciona una forma de mostrar pestañas horizontalmente. Cuando se usa junto con un ViewPager, un TabLayout puede proporcionar una interfaz familiar para navegar entre páginas en una vista deslizante. [53]

- ***ChipGroup:*** El ChipGroup está formado por varios Chips, que son elementos compactos que representan un atributo, texto, entidad o acción. [54] En la siguiente examen se puede observar el uso de este elemento en la aplicación.

Imagen 26: Ejemplo ChipGroup

Continuando con el diseño de la aplicación, en el caso de los donantes, se ha utilizado una barra de navegación inferior que permite el movimiento entre las tres principales partes: pantalla de inicio, el mapa y la lista de mapas. [55]

A continuación, se puede observar la barra mencionada anteriormente:

Imagen 27: Barra de navegación inferior donante

Otro elemento utilizado para la navegación, tanto en el caso del donante como en el caso del voluntario es el menú lateral, creado con un *Navigation Drawer*. De esta manera, el usuario puede acceder fácilmente a las pantallas de Donaciones, Configuración, Contacto, Ayuda, Sobre nosotros y Condiciones de uso. Además, puede ver la información principal del perfil y salir de la sesión. [51]

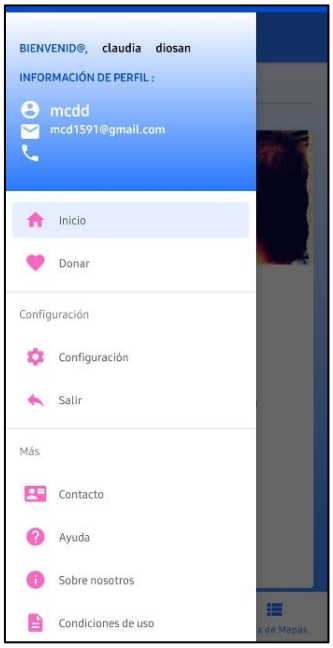


Imagen 28: Menú lateral usuario

En el caso del voluntario, para acceder a las pantallas principales, se utiliza un menú de botón de acción flotante a través del cual el usuario puede crear un nuevo perfil de persona necesitada o gestionar las entregas de donaciones. [30]

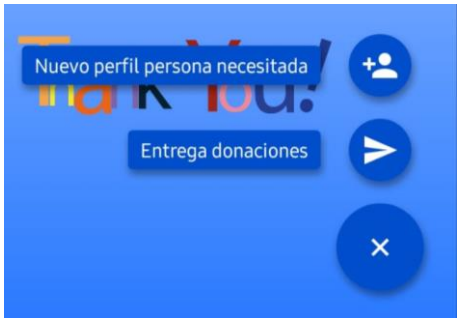


Imagen 29: Menú de botón de acción flotante

Por último, en el caso de cada perfil de persona necesitada, se puede gestionar simplemente seleccionando el ítem que se desea.

5.5. Diseño de pantallas y menús

En este apartado, se explicarán las pantallas de la aplicación en cuanto al diseño.

5.5.1. Pantallas de Intro

Es un conjunto de cuatro pantallas, creadas utilizando un *ViewPager* para deslizarse entre cada pantalla y que solo son visibles la primera vez que el usuario instala la aplicación. Estas pantallas se utilizan como pequeña introducción a la aplicación, explicando las funcionalidades principales.

Imagen 30: Pantallas de Intro

5.5.2. Pantalla de login

Esta es la pantalla inicial de la aplicación, que dispone de dos campos para introducir las credenciales de acceso a la cuenta (el correo y la contraseña) e iniciar sesión a través del botón de Inicio de Sesión. Aparte de esto, tenemos un texto que se puede presionar en el caso que el usuario ha olvidado su contraseña, un botón para acceder a crear una nueva cuenta y otro botón para entrar en la parte del Liquid Galaxy:

Imagen 31: Pantalla de login

5.5.3. Pantalla para recuperar la contraseña

Esta pantalla está compuesta por un campo para introducir el correo utilizado para crear la cuenta. En caso que el correo es válido, el usuario recibirá un correo con un link para poder cambiar su contraseña y en la pantalla se mostrará un dialogo de confirmación al través del cual el usuario puede volver a la pantalla de login:

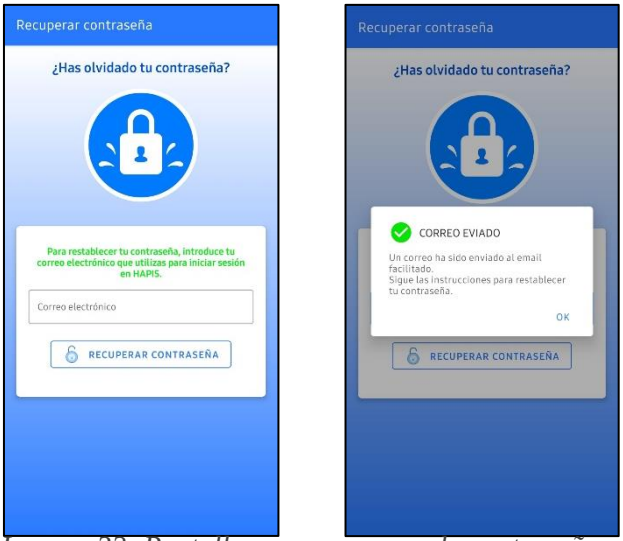


Imagen 32: Pantalla para recuperar la contraseña

5.5.4. Pantalla registro

Esta es una pantalla que aparece antes de que el usuario acceda al formulario de registro. Se utiliza para que el usuario pueda escoger si quiere crear el perfil como donante o como voluntario. Está formada por dos *MaterialCardView*, cada uno con dos botones: un botón que muestra un dialogo con información sobre que se puede hacer en caso de cada tipo de usuario y un botón para iniciar el registro. Un *MaterialCardView* no es más que una tarjeta para mostrar la información dentro.

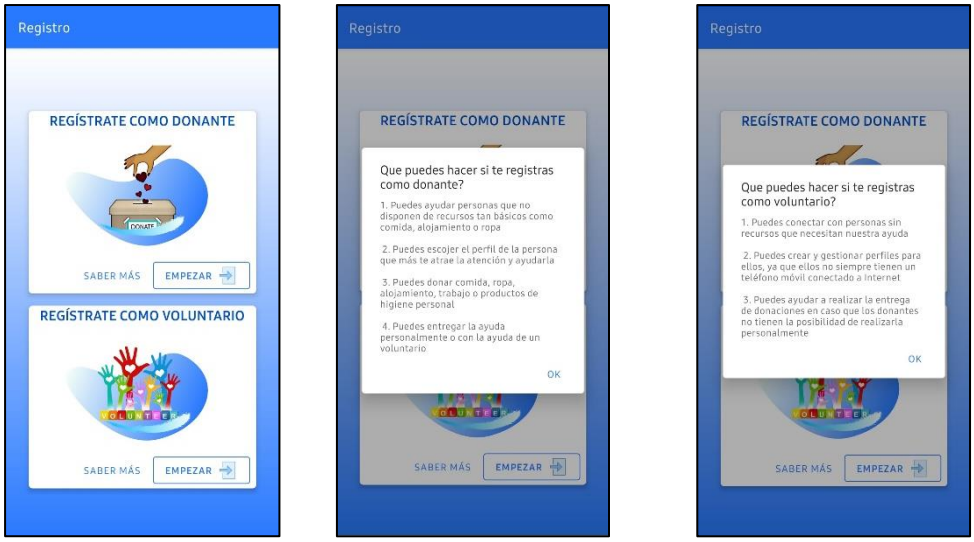


Imagen 33: Pantalla registro

5.5.5. Pantalla principal donante

La pantalla principal del donante está formada por un *RecyclerView* compuesto por los perfiles de las personas que necesitan ayuda. Al seleccionar un ítem se abrirá la pantalla de ayuda. En la pantalla también se puede encontrar un buscador que filtra la información de los perfiles por usuario, necesidad o dirección. Para la navegación, se utiliza un menú inferior para desplazarse entre las pantallas principales y un menú lateral para las pantallas menos utilizadas.

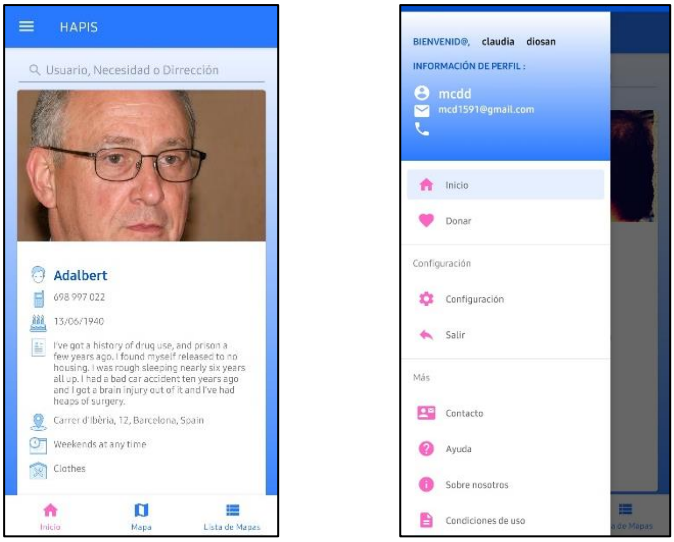


Imagen 34: Pantalla principal donante

5.5.6. Pantalla de ayuda

En esta pantalla el donante puede elegir que donación quiere realizar (ropa, comida, trabajo, alojamiento o productos de higiene personal) utilizando un *ChipGroup* y como quiere realizar la entrega (personalmente o a través de un voluntario) utilizando dos botones.

Imagen 35: Pantalla de ayuda

5.5.7. Ayuda personalmente

El usuario que escoge realizar la donación personalmente será dirigido a esta pantalla donde podrá ver un mapa con la ubicación de la persona que quiere ayudar. Además, se puede ver la dirección y el horario en el que puede encontrar a la persona necesitada en dicho lugar.

Imagen 36: Pantalla para ofrecer ayuda personalmente

5.5.8. Pantalla de ayuda a través de un voluntario

El usuario será dirigido a esta pantalla si elige que quiere realizar la donación a través de un voluntario. En esta página el donante podrá establecer un punto de encuentro con un voluntario para entregarle la donación. La pantalla está compuesta por un campo para escoger la ubicación y tres botones: dos botones para establecer la fecha y la hora y un botón para confirmar los datos. Una vez confirmados los datos, los voluntarios recibirán toda la información para ponerse en contacto con el donante.

Imagen 37: Pantalla para ofrecer ayuda a través de un voluntario

5.5.9. Mapa

Esta pantalla está compuesta por un mapa en el cual se pueden ver todos los marcadores de las personas necesitadas. Al pulsar un marcador, en la descripción saldrá el nombre de usuario y horario de la persona seleccionada. Pulsando la descripción, el usuario será directamente dirigido a la pantalla de ayuda.

Imagen 38: Mapa

5.5.10. Lista de mapas

La pantalla de lista de mapas está compuesta por un *RecyclerView* que contiene mapas de tipo *Lite*.

El modo lite muestra una imagen de mapa de bits correspondiente a un mapa de una ubicación determinada y con un nivel de zoom específico. Resulta útil cuando se desea proporcionar una serie de mapas en un flujo o un mapa demasiado pequeño como para admitir una interacción significativa. [56]

Cada mapa contiene también textos con el nombre de usuario de la persona necesitada y la ubicación, aparte de dos botones que abren la aplicación Google Maps: uno que muestra la ruta hasta la ubicación determinada y el otro que muestra solamente la ubicación.

Además, tenemos también un buscador que filtra los datos por usuario o por dirección.

Imagen 39: Lista de mapas

5.5.11. Pantalla principal voluntario

En esta pantalla se muestran los perfiles de personas necesitadas que ha creado el usuario registrado como voluntario. La primera vez que se inicia sesión, la pantalla no tendrá datos, ya que cada voluntario solo puede visualizar y editar los perfiles que ha creado. Para crear un nuevo perfil de usuario y para gestionar la entrega de donaciones, tenemos un menú de botón flotante.

Al pulsar un ítem, se abrirá la página para editar la información de perfil. En la pantalla también se puede encontrar un buscador que filtra la información de los perfiles por usuario, necesidad o dirección.

También se dispone de un menú lateral para las donaciones, la configuración, contacto, ayuda, sobre nosotros, condiciones de uso y para salir de la sesión.

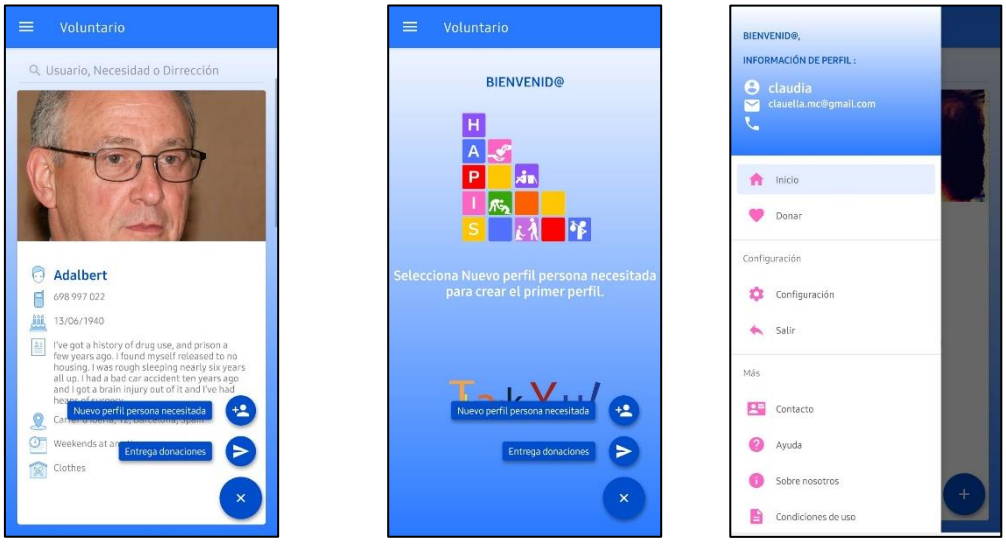


Imagen 40: Pantalla principal voluntario

5.5.12. Editar perfil persona necesitada

En esta pantalla se puede cambiar la foto de perfil, el teléfono, la historia de vida, la ubicación, el horario y la necesidad más importante. También existe un botón para borrar el perfil por completo.

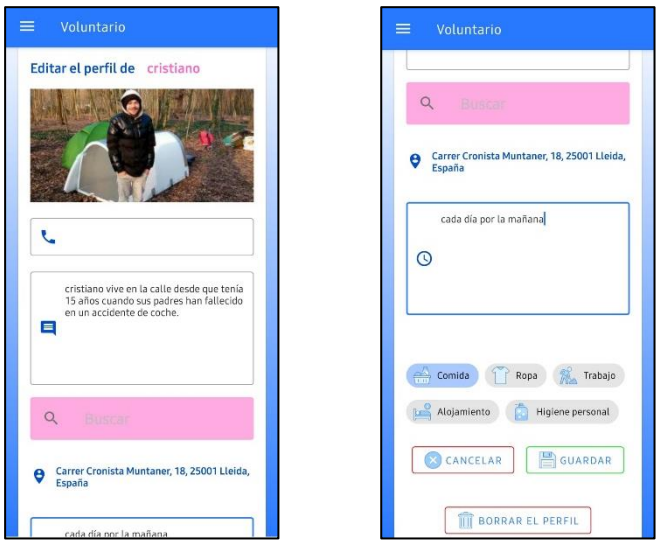
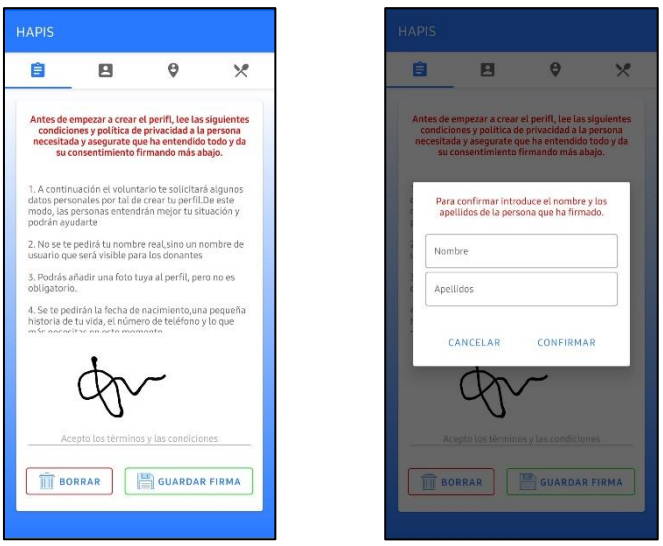


Imagen 41: Editar perfil persona necesitada

5.5.13. Crear nuevo perfil persona necesitada

En este caso, tenemos cuatro fragmentos. El primero, contiene los términos y las condiciones de privacidad que deberá leer la persona necesitada antes de que el voluntario proceda a la creación de su perfil. La pantalla contiene un cuadro para firmar y confirmar la firma con el nombre y los apellidos.

El segundo fragmento, contiene la foto de perfil y la información personal, como teléfono, cumpleaños y una pequeña historia de vida. A continuación, se completará la ubicación y, por último, un horario en el que la persona necesitada puede ser encontrada en la ubicación anterior y la necesidad más importante que tiene.



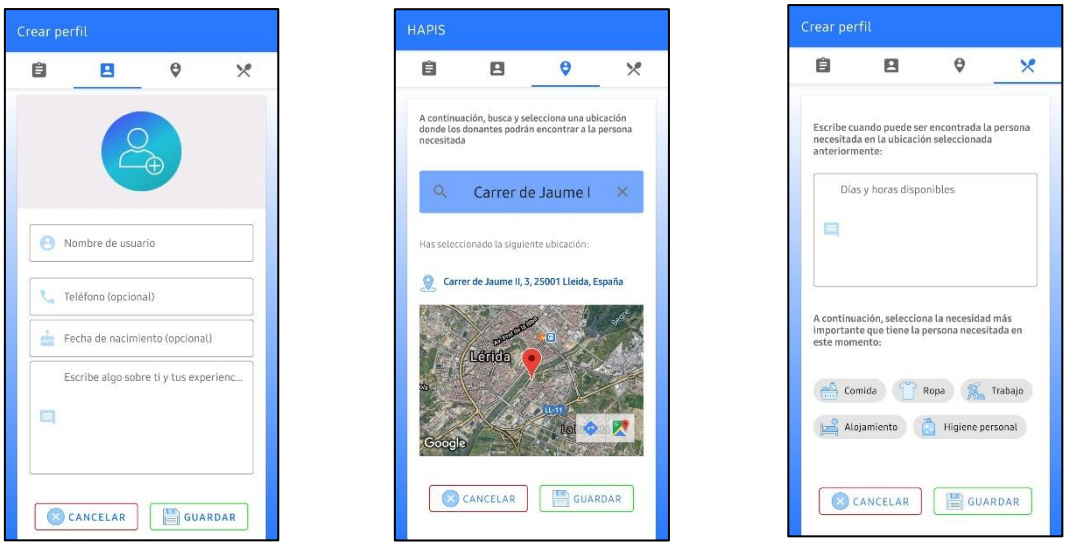
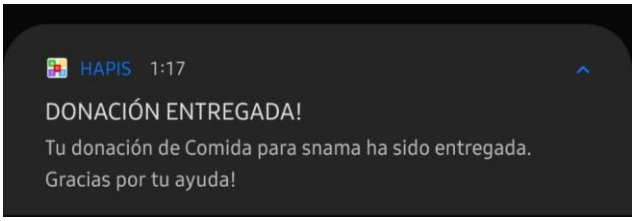


Imagen 42: Crear nuevo perfil persona necesitada

5.5.14. Entrega donaciones

En esta pantalla el voluntario puede ver la información necesaria para poder realizar una entrega de donación si quiere. En este caso, la información la reciben todos los voluntarios. En cada tarjeta con información se encuentra el nombre de usuario del donante, el tipo de donación que quiere realizar, la información de contacto del donante como teléfono y correo (el teléfono es opcional) y el punto de encuentro con la fecha y la hora. Como datos de la persona necesitada aparecen su nombre de usuario, ubicación y horario. Además, hay 3 botones:

- "Donante contactado" que se pulsa si el voluntario ha contactado y ha establecido un punto de encuentro con el donante. De este modo, el botón se desactiva y el donante aparece como "contactado" para que otros voluntarios no los contacten.
- "Cancelar entrega" que tiene dos opciones: cancelar toda la entrega y contactar de nuevo. La primera opción borra toda la información relacionada con esta entrega y la segunda opción pone el donante disponible de nuevo para que otro voluntario pueda contactarlo. Esta opción podría ser utilizada en caso que el voluntario que ha establecido el punto de encuentro en un primer momento ya no podría ir para recoger la donación.
- "Donación entregada" que marca la entrega como realizada y la borra de la página de entregas. En el momento que un voluntario marca una entrega como realizada, el donante recibe una notificación conforme su donación ha sido entregada.



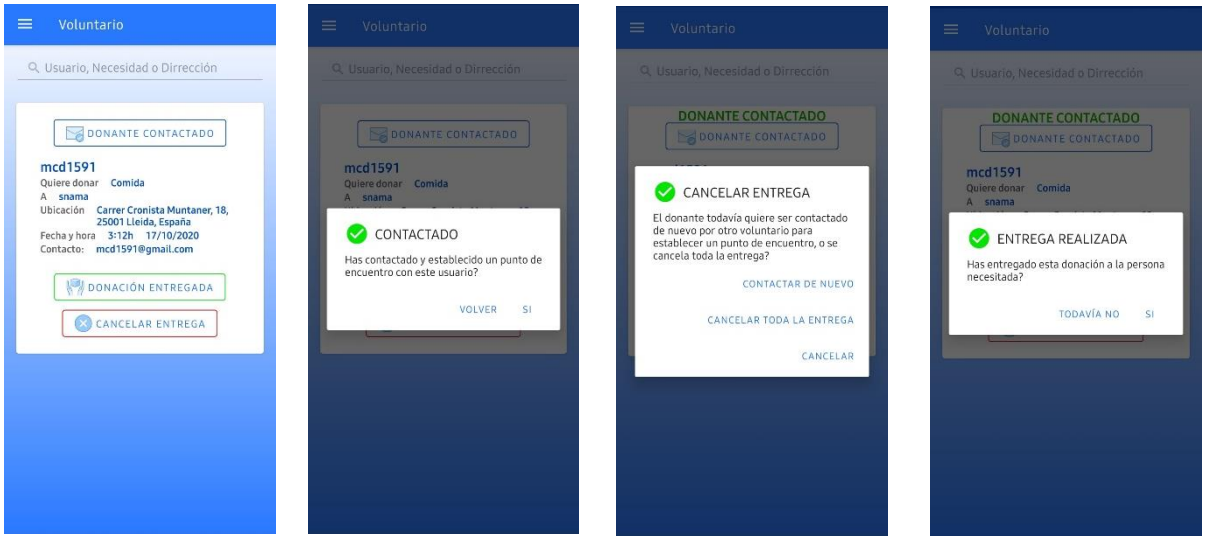


Imagen 43: Entrega donaciones

5.5.15. Donar

Esta pantalla es común a los dos tipos de usuario y contiene dos botones: un botón para realizar una donación de dinero pagando con la tarjeta de crédito o débito y un botón para mirar un video de 3 segundos aproximadamente para hacer una donación pequeña. En la pantalla de pago, se encuentra un campo para introducir la cantidad de dinero que se quiere donar y campos para completar los datos de la tarjeta. En la pantalla del video, simplemente se reproduce un video de YouTube.

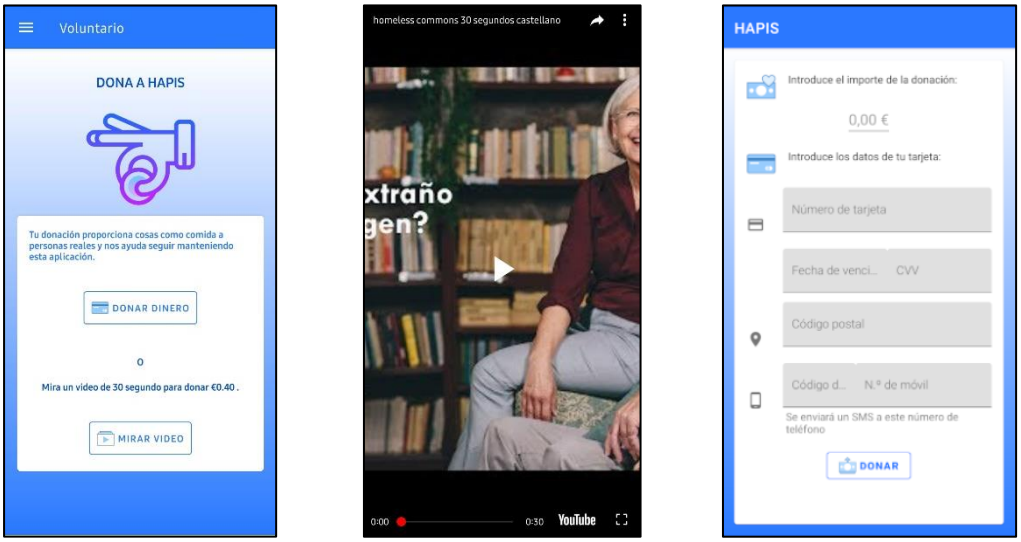


Imagen 44: Donar

5.5.16. Configuración

Esta es una pantalla de preferencias diferente en cada tipo de usuario:

- *Configuración donante:* el donante puede editar información de perfil como número de teléfono o cambiar su contraseña y puede borrar su cuenta definitivamente. También puede escoger si ver el mapa solamente cuando hay una conexión por Wifi. De este modo se ahorraría datos móviles.

- *Configuración voluntario:* el voluntario puede cambiar su número de teléfono y contraseña y puede escoger si en la página de entrega de donaciones quiere ver todas las entregas o solo las entregas para los perfiles creados por él. Este tipo de usuario no tiene la opción de borrar su cuenta, pero en la página de ayuda de la aplicación se menciona que, si quiere borrar la cuenta, debería ponerse en contacto con nosotros para asignar los perfiles creados por el a otros voluntarios y después proceder a eliminar la cuenta definitivamente.

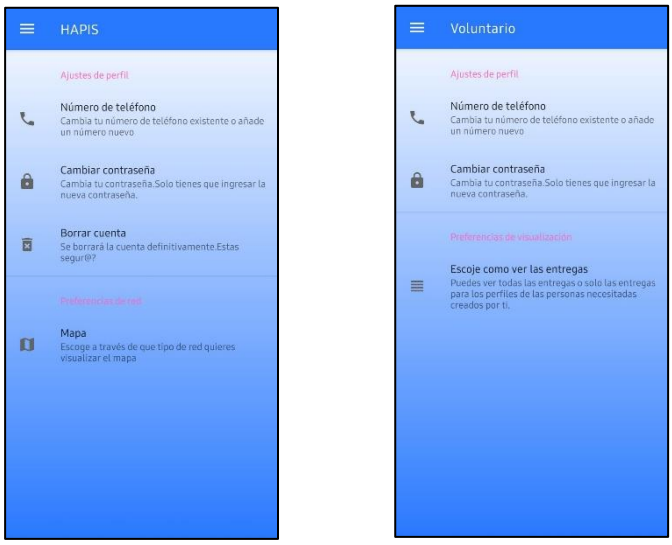


Imagen 45: Configuración

5.5.17. Contacto

Esta es otra pantalla común a los dos tipos de usuario en la cual hay dos campos para introducir el asunto y el mensaje y un botón para enviar.

Imagen 46: Contacto

5.5.18. Ayuda

En la pantalla de Ayuda el usuario puede encontrar información sobre las principales funcionalidades de la aplicación. El texto en esta pantalla cambia para cada tipo de usuario.

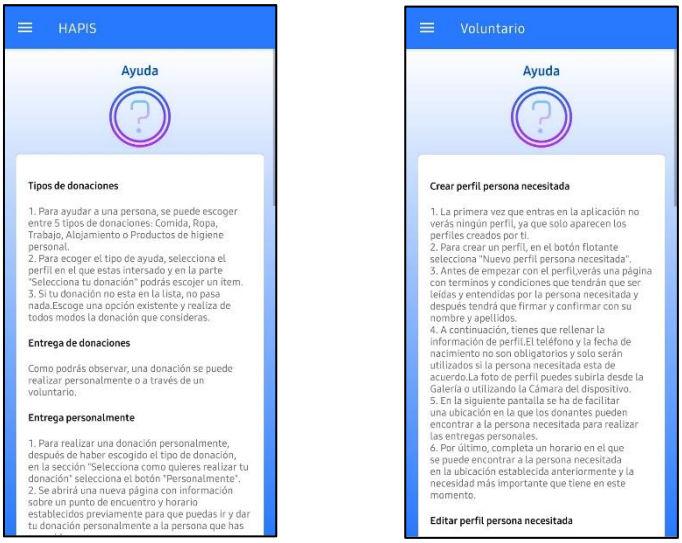


Imagen 47: Pantalla de ayuda

5.5.19. Sobre nosotros

Aquí se puede encontrar información sobre quien ha desarrollado la aplicación, donde encontrar e código y las principales redes sociales del proyecto. Esta página es común para los dos tipos de usuario.

Imagen 48: Pantalla Sobre nosotros

5.5.20. Condiciones de uso

En esta página, el usuario puede encontrar cuáles son sus obligaciones y sus derechos al utilizar esta aplicación. De este modo, puede consultarlas en cualquier momento.

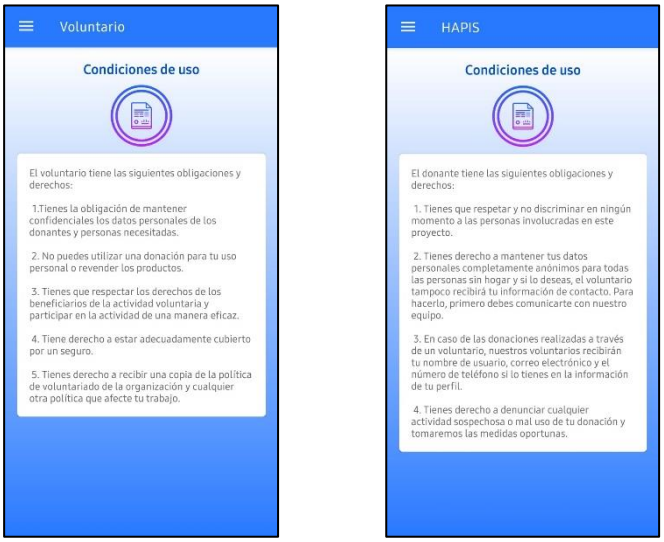


Imagen 49: Condiciones de uso

5.5.21. Pantalla principal Liquid Galaxy

Esta pantalla está compuesta por tres botones: un botón para acceder a las ciudades y dos botones para las estadísticas. En caso de las estadísticas por ciudad, en el Liquid Galaxy se realizará un tour por cada ciudad, esperando aproximadamente 20 segundos en cada una y mostrando las estadísticas locales en una pantalla. Si se selecciona Estadísticas globales, en el Liquid Galaxy se mostrará la tierra y una pantalla con las estadísticas globales.

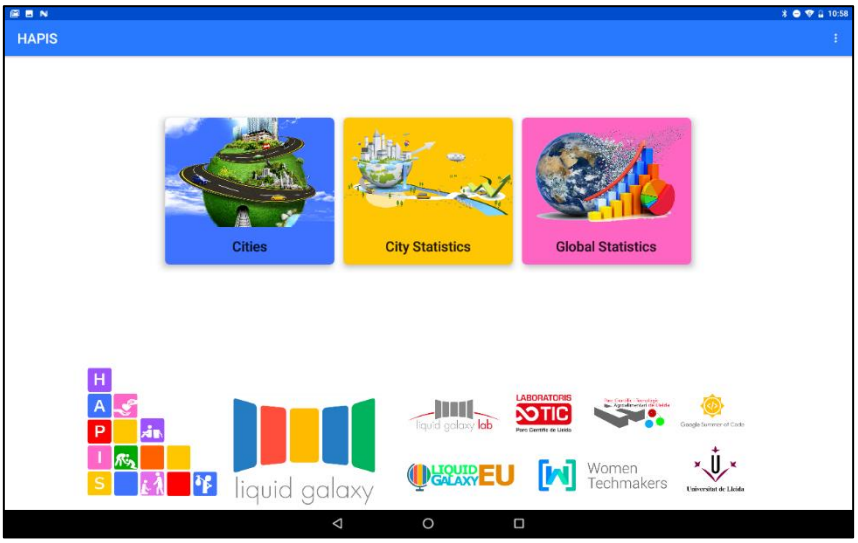


Imagen 50: Pantalla principal parte Liquid Galaxy

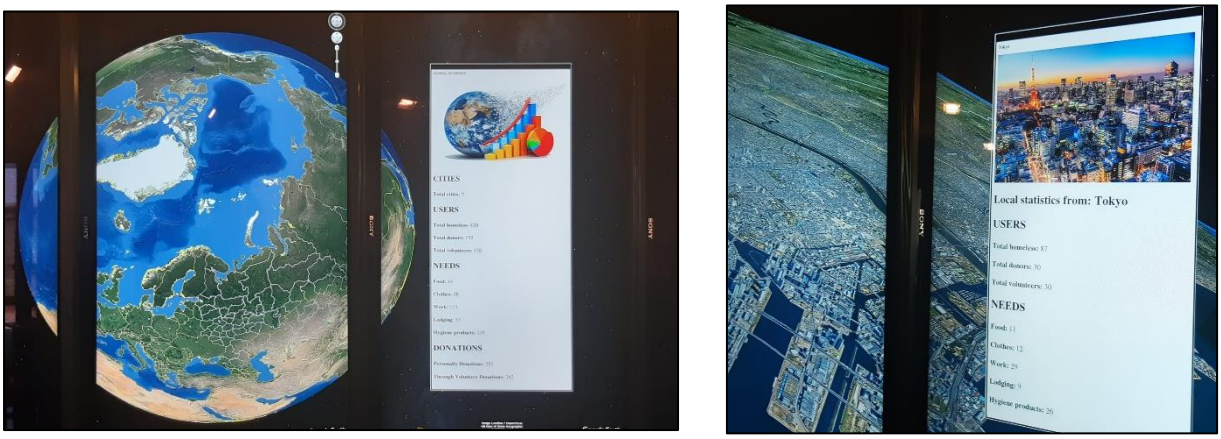


Imagen 51: Estadísticas globales y estadísticas por ciudad en el Liquid Galaxy

5.5.22. Ciudades

En esta pantalla se muestran todas las ciudades en las cuales existen personas necesitadas, donantes o voluntarios registrados. Al seleccionar una ciudad, en la pantalla del Liquid Galaxy se realiza un viaje a la misma. En la aplicación se abrirá una pantalla con más opciones para la ciudad. También hay un buscador para filtrar las ciudades.

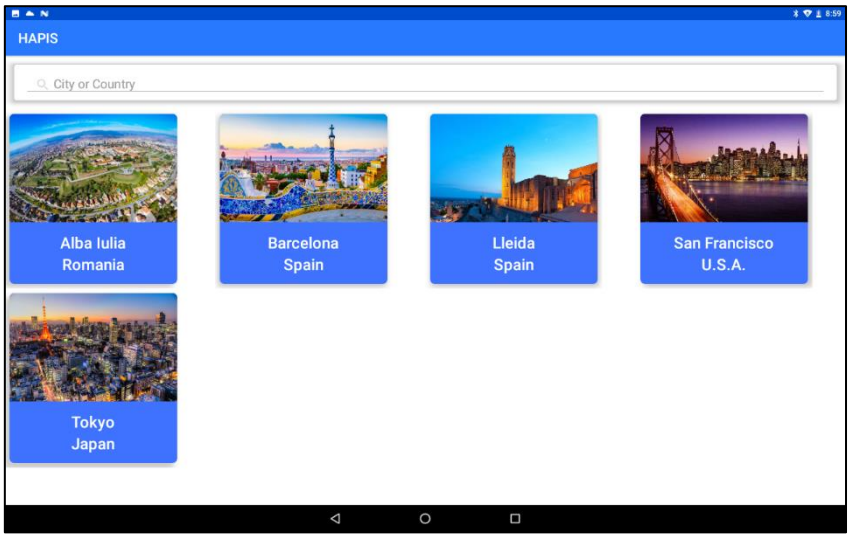


Imagen 52: Ciudades

5.5.23. Pantalla para una ciudad

Aquí se pueden encontrar cuatro botones: un botón para cada perfil de usuario que abrirá una lista de todos los usuarios de ese tipo y un botón, llamado Live Overview que se muestra en el Liquid Galaxy, situándose sobre la ciudad, mostrando todos los marcadores de las personas necesitadas, en una pantalla las estadísticas locales y en otra pantalla la información sobre cada persona necesitada, esperando cada 20 segundos entre cada perfil. Además, realiza un movimiento de rotación entorno a la ciudad.

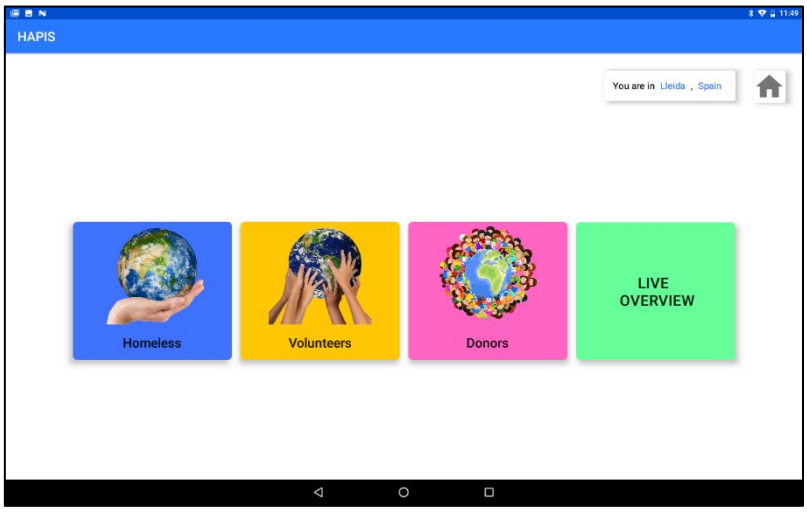


Imagen 53: Pantalla ciudad

Imagen 54: Live Overview visto en el Liquid Galaxy

5.5.24. Lista usuarios

En esta pantalla aparece una lista de todos los usuarios de un determinado tipo (persona necesitada, voluntario o donante). Cada usuario se encuentra en un botón que, si se pulsa, en el Liquid Galaxy se realiza un viaje en la ubicación de esta persona y se muestra una ventana con información básica. Cada botón tiene 3 botones más: un botón que muestra información extra, un botón que muestra las transacciones y un botón que realiza una órbita entorno a este usuario en el Liquid Galaxy.

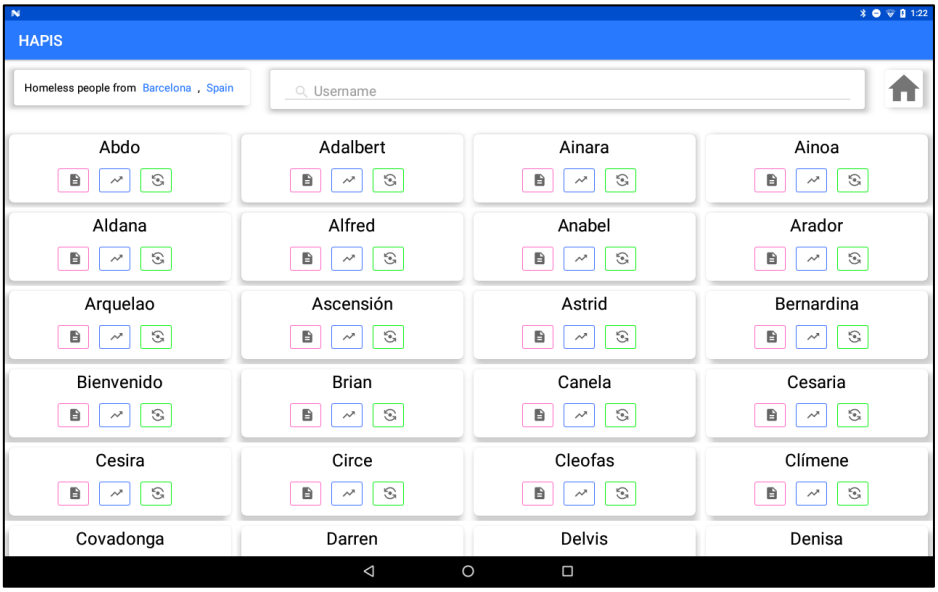


Imagen 55: Pantalla con la lista de usuarios

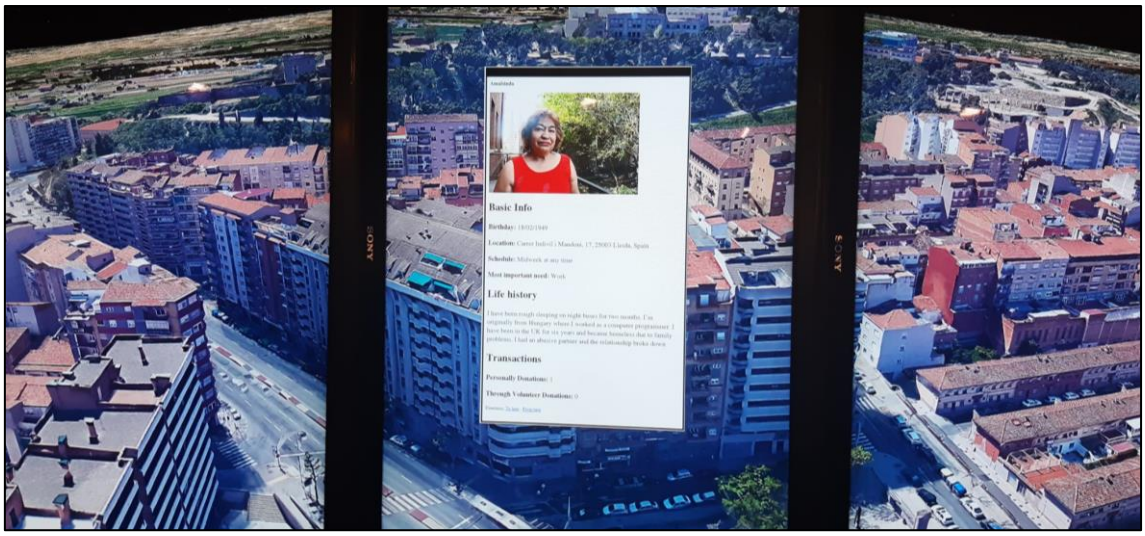


Imagen 56: Ventana de información persona necesitada Liquid Galaxy

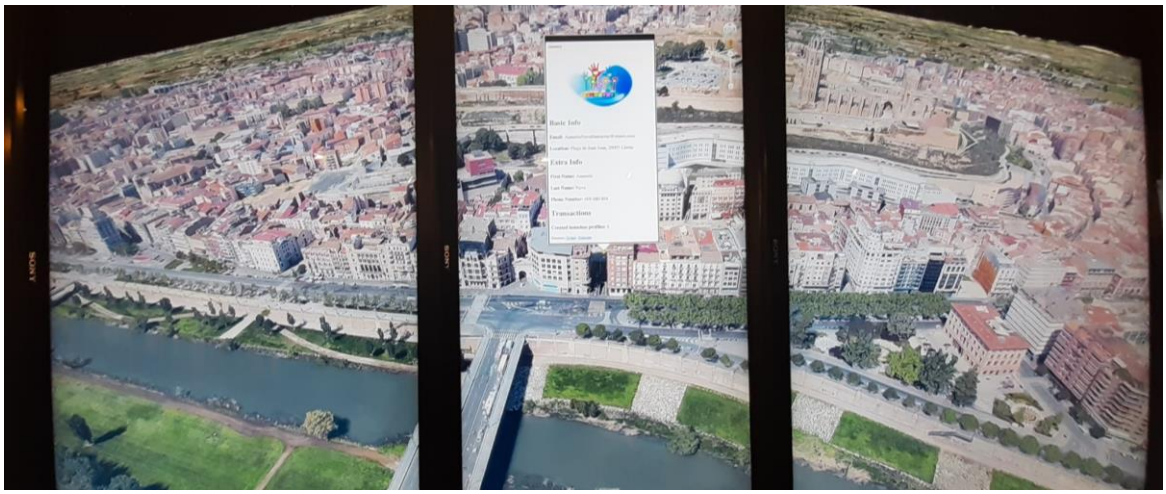


Imagen 57: Ventana de información voluntario Liquid Galaxy

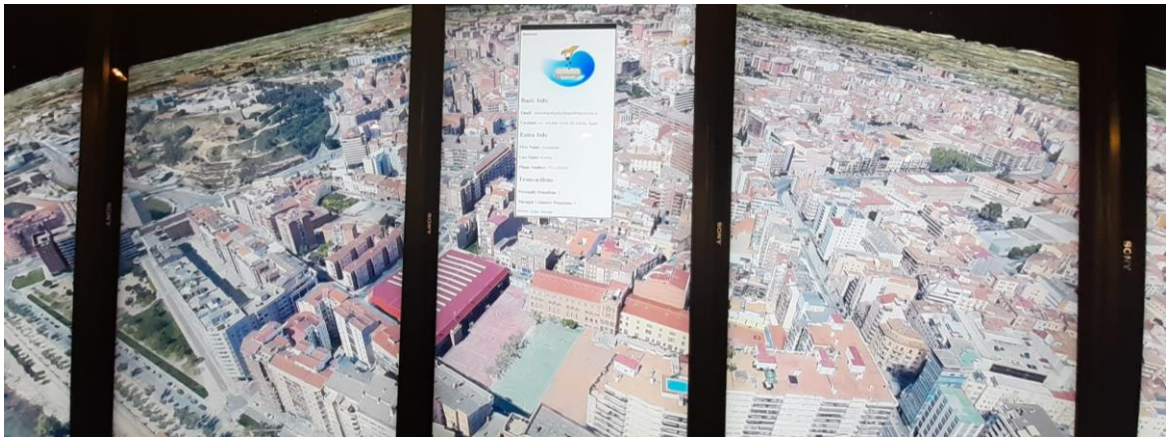


Imagen 58: Ventana de información donante Liquid Galaxy

5.5.25. Pantalla configuración

A esta pantalla se accede desde el menú que se encuentra en la pantalla inicial de la parte del Liquid Galaxy. La pantalla contiene las preferencias relacionadas con la conexión con el Liquid Galaxy, como usuario, contraseña, IP y puerto del master. Además, cuenta con unas opciones avanzadas donde el usuario puede escoger en que pantalla quiere ver las diferentes ventanas con información.

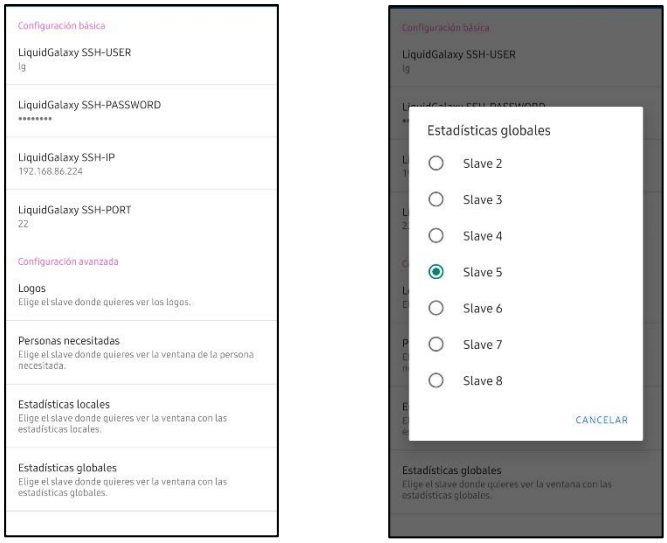


Imagen 59: Pantalla configuración Liquid Galaxy

5.5.26. Herramientas

Esta pantalla también es parte del menú y contiene cuatro botones para controlar el Liquid Galaxy: limpiar los archivos. kml, relanzar, reiniciar y apagar el Liquid Galaxy. Además, cuenta con un menú a través del cual se tienen que instalar los requisitos la primera vez que se utiliza la aplicación con un sistema Liquid Galaxy. También se puede desinstalar toda la información relacionada con la aplicación, pulsando el botón Desinstalar. Se tiene que comentar que estos dos botones funcionan si la aplicación ya está correctamente conectada al Liquid Galaxy.

Imagen 60: Pantalla de herramientas

5.5.27. Pantalla de ayuda

En esta pantalla se puede encontrar información sobre todo el funcionamiento de la aplicación, relacionado con el Liquid Galaxy. También se explica cada pantalla y cada botón.

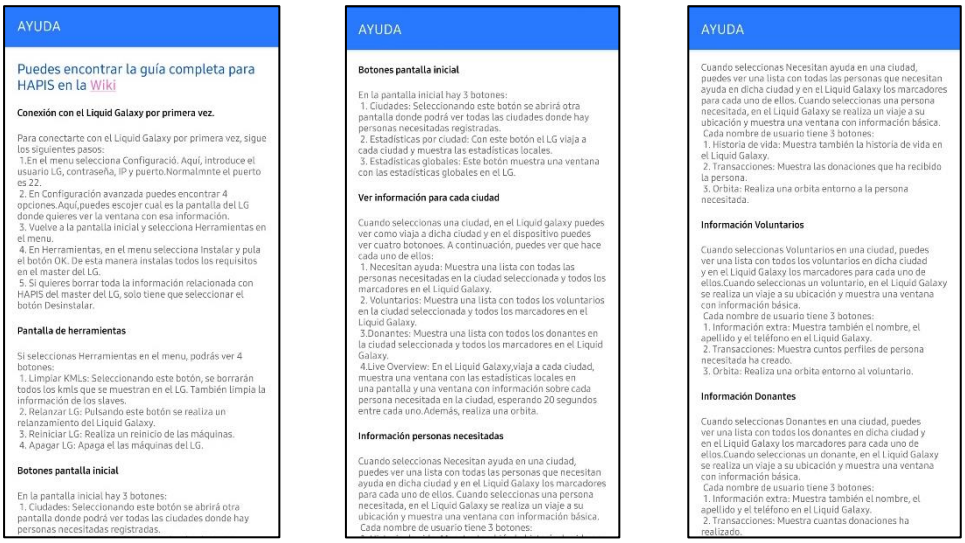


Imagen 61: Pantalla de ayuda

5.5.28. Pantalla Sobre Nosotros

Esta pantalla contiene información sobre el desarrollo de la aplicación, las personas que han ayudado y la fuente de todas las fotos e iconos presentes en las diferentes pantallas.

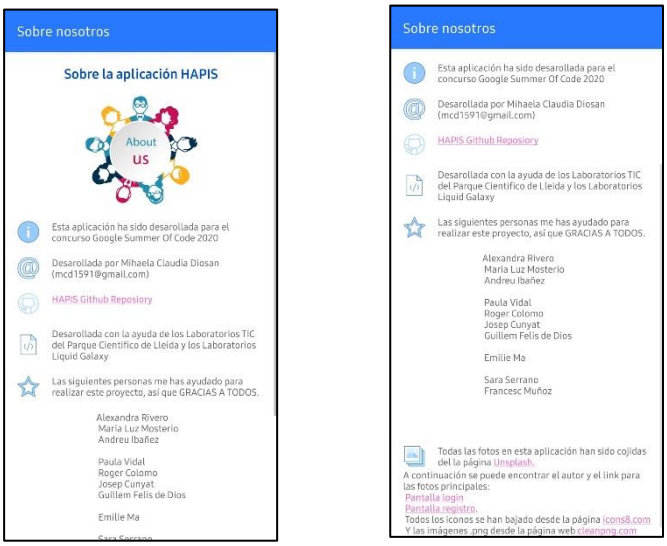


Imagen 62: Pantalla Sobre Nosotros

5.6. Iconos y logo

Tanto el logo de la aplicación, como el icono han sido creados por mí. El logo de la aplicación se muestra mientras se carga la aplicación (en el Splashscreen).

Para cambiar el icono de la aplicación ha sido necesario cambiar el fichero *AndroidManifest.xml*.

Primero, se ha abierto las carpetas que contienen el icono por defecto y se han reemplazado los ficheros con el icono deseado. Hay que comentar que la carpeta que contiene el icono por defecto, se llama *mipmap* y se introdujo a partir de la versión 1.1. de Android Studio. Cuando creamos un proyecto en Android Studio, por defecto, ya nos coloca ahí el icono con el nombre de *ic_launcher*. [57] Las diferentes densidades de los iconos se han generado a través de una página web. [58]

Después de esto, en el fichero *AndroidManifest.xml* se han cambiado las siguientes líneas, donde *hapis_icon* y *hapis_icon_round* son los dos iconos generados (redondo y cuadrado):

```
android:icon="@mipmap/hapis_icon"
```

```
android:roundIcon="@mipmap/hapis_icon_round"
```

Para añadir el logo en el Splashscreen, primero se ha diseñado el layout[16] dentro de la carpeta drawable. [59]

De esta manera, ponemos nuestra imagen en el centro de la pantalla:

```
<layer-list xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:drawable="@color/white"/>
    <item>
        <bitmap
            android: gravity="center"
            android:src="@drawable/splash_image"
            />
    </item>
</layer-list>
```

Luego definimos un tema al cual le asignamos como ***windowBackground*** la drawable/layout que acabamos de crear. En este caso, este tema se utilizará para las pantallas de Intro.

[16] Los layouts son elementos no visuales destinados a controlar la distribución, posición y dimensiones de los controles que se insertan en su interior. [85]

```
<style name="IntroTheme" parent="AppTheme">
    <item name="colorPrimary">@color/white</item>
    <item name="colorPrimaryDark">@color/white</item>
    <item name="colorAccent">@color/white</item>
    <item name="android:windowBackground">@drawable/splash</item>
</style>
```

Para asignar el tema a la actividad que contiene la Intro, se ha modificado el contenido del archivo *AndroidManifest.xml*.

```
<activity
    android:name=".intro.IntroActivity"
    android:theme="@style/IntroTheme">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Por último, en la *IntroActivity* se configura el tema que hemos creado, con el splashscreen:

```
protected void onCreate(Bundle savedInstanceState) {
    setTheme(R.style.AppTheme);
    super.onCreate(savedInstanceState);

...
}
```

Finalmente, cuando se carga la aplicación, se puede ver la siguiente imagen con el logo incluido:

Imagen 63: SplashScreen

5.6.1. Fuentes fotos, iconos e imágenes

Los iconos utilizados en la aplicación en los botones, a principio de algunas líneas de texto o en algún campo para introducir datos se han bajado de una misma página web, intentando mantener siempre el mismo diseño. [60]

En cuanto a las imágenes con fondo transparente de tipo .png, también se ha utilizado la misma página web, que contiene imágenes gratuitas. [61]

Las fotos utilizadas en algunas pantallas, como la pantalla de login, o las pantallas de registro se han bajado de la página web Unsplash que contiene imágenes bajo licencia libre. [62]

Por último, todas las fotos que se encuentran como imagen de perfil de las personas necesitadas, se han bajado de diferentes bases de datos y páginas web. [63]

6. Implementación

Tal como se ha comentado anteriormente, el proyecto se ha desarrollado utilizando el entorno de desarrollo integrado oficial para la plataforma Android, Android Studio. Como lenguaje de programación se ha utilizado Java y para la parte de backend se ha utilizado Firebase. Por último, como parte extra, se ha utilizado el sistema Liquid Galaxy.

6.1. Estructura del proyecto

A continuación, se mostrará un resumen de la estructura del proyecto en Android Studio y qué contiene cada directorio, dividiendo todo en tres partes principales:

- *manifests:* contiene el archivo *AndroidManifest.xml*.
- *java*: contiene los archivos de código fuente Java.
- *res:* contiene todos los recursos sin código, como diseños XML, strings de IU e imágenes de mapa de bits. [38]

6.1.1. Archivo AndroidManifest.xml

En el archivo *AndroidManifst.xml*, se guarda la siguiente información: [65]

- ***Nombre del paquete e ID de la aplicación:*** El elemento raíz del archivo de manifiesto contiene un atributo para el nombre del paquete de la aplicación, que coincide con la estructura del directorio del proyecto.

- ***Permisos:*** La aplicación debe solicitar permiso para acceder a datos del usuario confidenciales o a determinadas funciones del sistema (como la cámara y el acceso a Internet). Cada permiso se identifica con una etiqueta única. En este caso, se necesitan permisos al almacenaje del sistema, para cargar la foto de perfil de la persona necesitada, permisos para el internet y para saber el estado de la red (utilizadas por ejemplo para saber cuándo el dispositivo tiene conexión a Internet o qué tipo de conexión utiliza) y, por último, permisos para acceder a la localización utilizados para las páginas de mapas.

- ***Íconos:*** Algunos elementos del manifiesto tienen atributos *icon* y *label* para mostrar un pequeño ícono y una etiqueta de texto, respectivamente, a los usuarios según el componente de aplicación correspondiente. El icono de la aplicación se muestra según la explicación del capítulo anterior.

- ***Actividades:*** Para declarar el componente de una actividad se utiliza `<activity>`. Para cada actividad declarada, aparte del nombre utilizamos el atributo *label*, mencionado anteriormente para mostrar el nombre de la actividad

y en algunos casos los atributos *configChanges* y *screenOrientation* para determinar la orientación de la actividad en la pantalla, ya que por motivos de diseño algunas pantallas están orientadas de manera vertical.

- ***Uses-permission:*** Especifica un permiso de sistema que debe conceder el usuario para que la aplicación funcione correctamente. En este caso se ha utilizado para declarar la clave API de Maps SDK para un correcto funcionamiento de los mapas y para incorporar la versión de Servicios de Google Play con la que se compiló la aplicación. [65]

6.1.2. Java

A continuación, se puede observar la estructura principal de la parte del código java del proyecto:

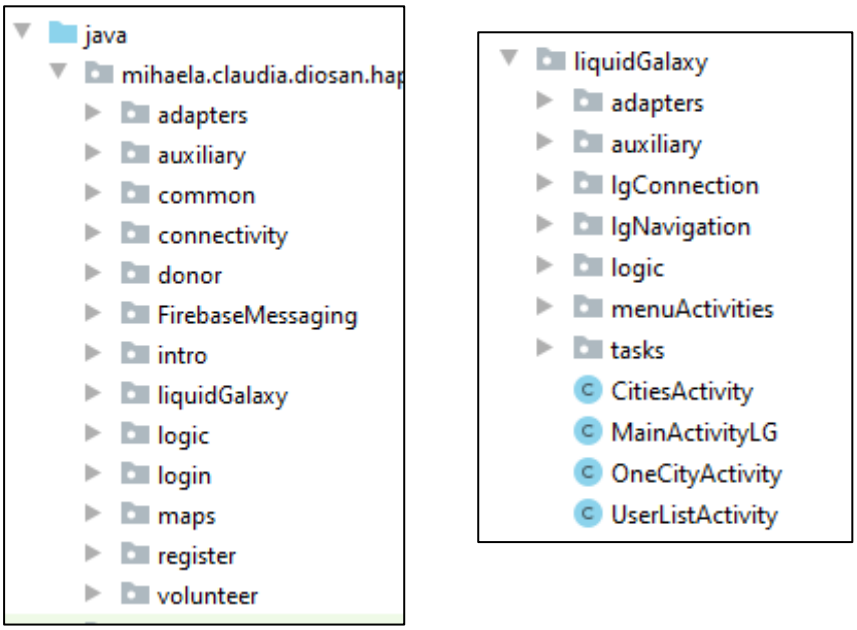


Imagen 64: Estructura proyecto parte Java

- ***adapters:*** En este paquete se encuentra las clases adaptador para los fragmentos principales de los donantes y de los voluntarios (*HomelessAdapter)* y la clase *DeliveryAdapter* para el fragmento en el que salen las entregas que tienen que realizar los voluntarios.
- ***auxiliary:*** Contiene una clase llamada *Help* en la cual se encuentran métodos auxiliares que pueden ser reutilizados en varias clases.
- ***common:*** Contiene los fragmentos y actividades comunes a los dos tipos de usuario, como la clase para realizar donaciones, la configuración, el contacto, sobre nosotros o las condiciones de uso.
- ***connectivity:*** Contiene la clase necesaria para comprobar el estado y el tipo de conexión a Internet.

- ***donor:*** Contiene las actividades y los fragmentos principales relacionados con el tipo de usuario donante.
- ***FirebaseMessaging:*** Contiene la clase *MyFirebaseMessagingService*, encargada de gestionar las notificaciones push cuando un voluntario marca una entrega como realizada.
- ***intro:*** Contiene las clases necesarias para mostrar las pantallas de introducción cuando la aplicación se instala por primera vez.
- ***liquidGalaxy:*** Contiene la implementación de toda la parte relacionada con el Liquid Galaxy.
- ***logic:*** Contiente las clases con la estructura principal de las personas que necesitan ayuda y las entregas.
- ***login:*** Contiene las clases que definen la pantalla de login y la pantalla utilizada en caso que se olvida la contraseña.
- ***maps:*** Contiene la implementación para las dos pantallas principales de mapas del tipo de usuario donante.
- ***register:*** Contiene la implementación de las dos actividades de registro.
- ***volunteer:*** Contiene las actividades y los fragmentos principales relacionados con el tipo de usuario voluntario.

6.1.3. Recursos de la aplicación

En nuestro caso, utilizamos los siguientes recursos:

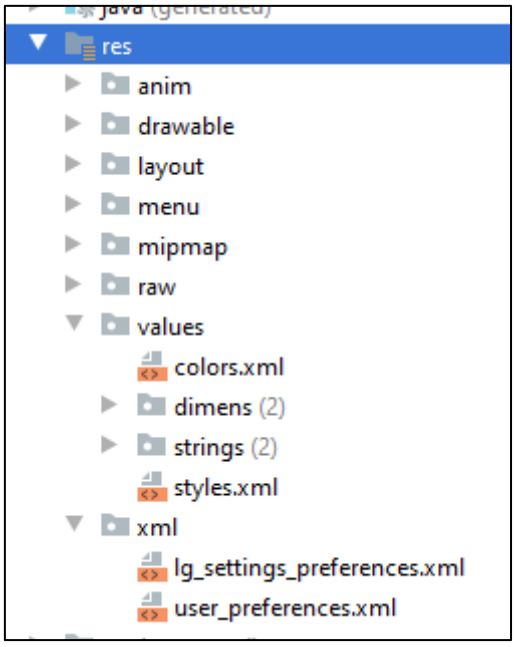


Imagen 65: Estructura proyecto parte recursos

- ***anim/:*** Contiene archivos XML que definen animaciones de interpolación de movimiento. Utilizado para el movimiento del botón *Start* de las pantallas de Intro y para el movimiento entre algunas pantallas.
- ***drawable/:*** Archivos de mapas de bits (.png, .9.png, .jpg y .gif) o archivos XML utilizados como elementos de diseño.
- ***mipmap/:*** Archivos de elementos de diseño para diferentes densidades de los íconos de selectores. En este caso se ha utilizado para guardar los iconos de la aplicación.
- ***values/:*** Archivos XML que contienen valores simples, como strings, valores enteros y colores. Para los strings se han utilizado los idiomas inglés y español y para las dimensiones se han utilizado dos archivos: normal y large en caso de la Tablet.
- ***xml/:*** Aquí se han guardado los archivos de configuración: uno para la parte de usuarios y uno para la parte del Liquid Galaxy.

6.2. Decisiones de implementación

6.2.1. Base de datos Firebase

Se ha tomado la decisión de utilizar como backend la base de datos Firebase, ya que es una herramienta fácil de utilizar e integrar en la aplicación, ofreciendo numerosos beneficios como son la escalabilidad o la seguridad. Teniendo en cuenta que de momento la aplicación se encuentra en una primera etapa y será utilizada con datos de prueba, también se ha de mencionar que será todo gratuito.

6.2.2. Autenticación

En este caso se utiliza la autenticación basada en correo electrónico y contraseña. En un principio, tenía planeado utilizar también el método de autenticación con Google, pero a la hora de la implementación me he dado cuenta que no es posible porque en el registro del usuario se decide qué tipo de perfil quiere crear: donante o voluntario y a través de la autenticación con Google sería imposible tomar esta decisión.

6.2.3. Orientación

Como se podrá ver, algunas pantallas de la aplicación utilizan siempre la orientación vertical. Esta decisión se ha tomado después de constatar que la información no quedaba bien en la pantalla si estábamos con la orientación horizontal. Las actividades en las cuales se ha utilizado esta opción son las actividades principales del donante y del

voluntario y la actividad de pago. Para hacer esto, se ha tenido que añadir los siguientes atributos en el archivo *AndroidManifest.xml*, en la actividad que necesitamos cambiar:

```
android:configChanges="orientation|screenSize"
android:screenOrientation="portrait"
```

android:configChanges enumera los cambios de configuración que controlará la actividad. Cuando se produce un cambio de configuración en tiempo de ejecución, la actividad se cierra y se reinicia de forma predeterminada. Sin embargo, si se declara una configuración con este atributo, la actividad no se reiniciará. Se utiliza con los atributos *orientation* y *screenSize* para detectar cuando el usuario gira el dispositivo.

screenOrientation utilizado con *portrait* define la orientación en la que se mostrará la actividad en el dispositivo, en este caso en modo vertical.

6.2.4. Reutilización de código

Para obtener un mejor rendimiento de la aplicación y un código fácil de entender, se ha decidido crear algunas clases con métodos que serán utilizados en varias otras clases y así no tener que volver a escribir el mismo código

En general se utiliza herencia para conseguir reutilizar el código. La herencia es cuando un objeto se basa en otro objeto o clase, usando la misma implementación o comportamiento. Esto es un mecanismo para la reutilización de código para permitirnos extensiones independientes del software original mediante clases públicas e interfaces. [66]

La clase *Help.java* contiene métodos para mostrar los avisos, las ventanas de alerta o para validar los campos en los que se introduce texto como el teléfono, el usuario o diversos campos utilizados para crear el perfil de usuario o de la persona necesitada. Estos métodos se reutilizan en varias actividades y fragmentos.

También hay varias actividades comunes que se utilizan para los dos tipos de usuario. En este caso el archivo que define el *layout* es el mismo, pero los elementos como texto, imágenes o botones se cambian programáticamente en la clase java por tal de poder reutilizar el mismo diseño de una página y no tener que repetir el mismo código.

La clase *POIController.java* contiene los métodos principales para enviar y mostrar la información en el Liquid Galaxy y se utiliza en casi todas las demás clases sin tener que repetir estos métodos.

También hay una clase de ayuda en la parte del Liquid Galaxy que contiene todos los métodos necesarios para poder obtener las estadísticas globales y locales.

Otra manera de reutilizar código es en el diseño de las pantallas es incluir un *layout* entero. En este caso se ha utilizado para añadir el *header*[17] en el menú lateral de cada

[17] El header es la cabecera del menú desplegable.

tipo de usuario. Para hacer eso, se ha creado aparte el diseño del *header* y luego se ha incluido en el elemento *NavigationView* de cada actividad principal de la siguiente manera:

```
app:headerLayout="@layout/user_header"
```

Teniendo este tipo de clases reutilizables también hace el trabajo mucho más fácil si en un futuro se quiere crear una nueva versión de la aplicación, incluido mejoras.

6.2.5. Gestión de recursos

La aplicación se encuentra disponible en dos idiomas: inglés y castellano. Esta decisión se ha tomado porque para el programa Google Summer Of Code era necesario realizarlo en inglés. Por otro lado, teniendo en cuenta que en un futuro podríamos publicar la aplicación en España se ha decidido incorporar también el castellano. En caso que la aplicación se publica, tengo pensado añadir también el catalán como idioma, teniendo en cuenta que seguramente primero funcionaría en Lleida.

Para conseguir que el idioma de la aplicación cambie al cambiar el idioma del dispositivo móvil, en la carpeta *values/strings* se ha añadido un fichero de tipo .xml por cada idioma.

Para añadir un nuevo idioma, primero, hacemos click derecha en la carpeta *values* y seleccionamos *New->Values resource file*:

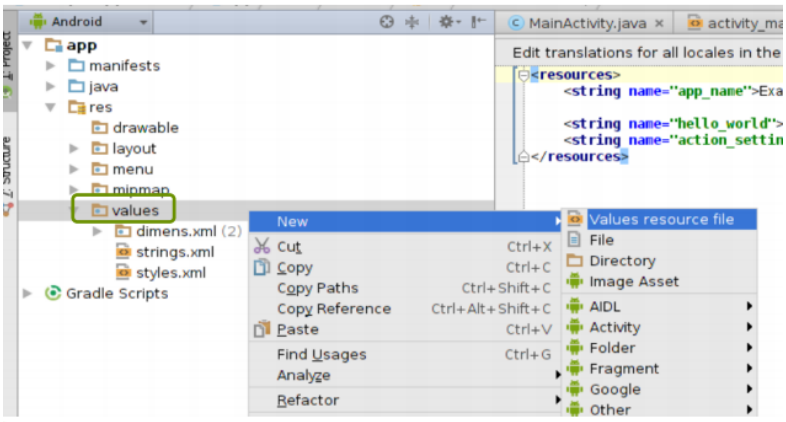


Imagen 66: Crear nuevo fichero strings.xml parte 1

Después, como nombre ponemos *strings* y seleccionamos como calificador disponible *Locale* y el idioma que queremos. Después, simplemente tenemos que traducir las frases en el nuevo fichero.

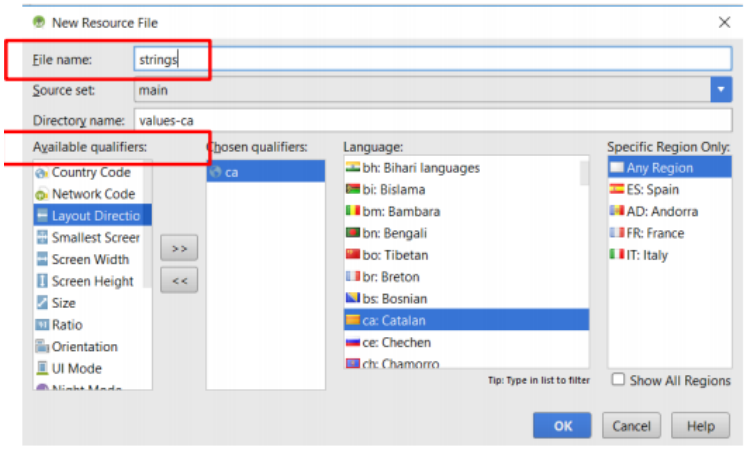


Imagen 67: Crear nuevo fichero strings.xml parte 2

También se han creado los archivos .xml tanto para la versión normal del dispositivo, como tipo *large* que se utiliza en caso de una Tablet y en algunos casos se ha creado un diseño distinto cuando el dispositivo tiene una orientación horizontal. Esta decisión se ha tomado considerando que tenemos también la parte del Liquid Galaxy y la aplicación seguramente se utilizará también en dispositivos de tipo Tablet. Por otro lado, hay que comentar que todas las dimensiones de los elementos que se encuentran en los *layouts* se encuentra en dos ficheros llamados *dimens.xml* y *dimens.xml(large)*. De esta manera resulta mucho más fácil a la hora de cambiar el tamaño de un elemento, ya que solo hay que cambiarlo en este fichero y se aplicará en todos los sitios en los cuales lo hemos utilizado. El procedimiento en estos dos casos es similar al de añadir un nuevo fichero *strings.xml*.

Para añadir un nuevo fichero *layout* hacemos click derecha en la carpeta *layout* y seleccionamos *New-> Layout Resource File*. Como nombre de fichero ponemos el nombre del archivo .xml que queremos ver en modo horizontal o en tamaño *large*. Si queremos añadir un archivo para el modo horizontal, escogemos como calificador disponible *Orientation->Landscape* y si queremos añadir un archivo para el tamaño, escogemos *Size->Large.*

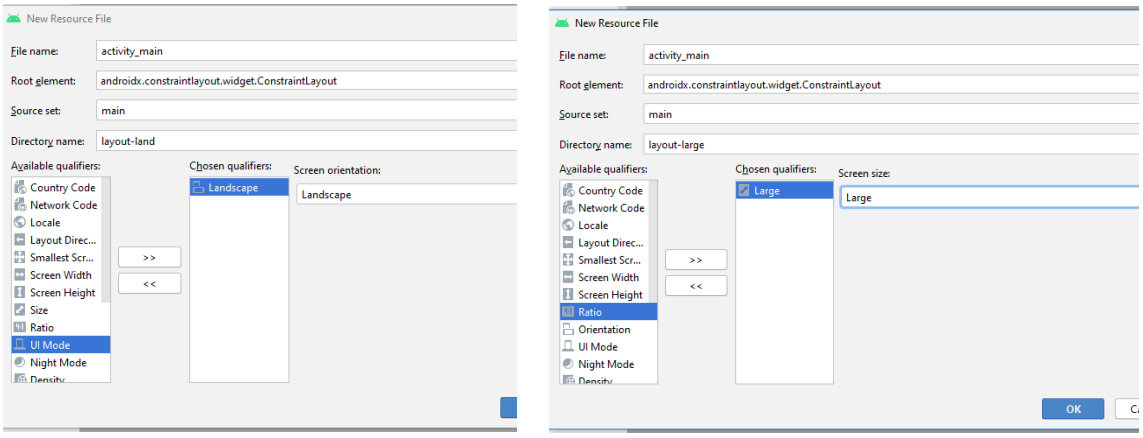


Imagen 68: Crear layout horizontal o tamaño para la tablet

En caso de añadir las dimensiones, creamos los archivos *dimens.xml* siguiendo el procedimiento descrito anteriormente.

6.2.6. Añadir la librería Android Image Cropper

En una primera versión de la aplicación, cuando un voluntario creaba o editaba un perfil para una persona necesitaba, simplemente tenía la opción de seleccionar la foto que quería añadir en la galería.

Después de realizar las pruebas con usuarios y recibir varias opiniones negativas conforme no se puede sacar una foto utilizando la cámara del teléfono, o que la imagen no se puede ajustar he decidido añadir la librería Android Image Cropper [35]. De este modo, el usuario puede escoger si quiere sacar la foto con la cámara del dispositivo o si quiere subirla desde la galería. Además, antes de confirmar puede cortar o girar la imagen.

Por otro lado, también he realizado un cambio en la parte del Liquid Galaxy. En la página *Ciudades* cada tarjeta contiene una foto de la ciudad. En una primera versión de la aplicación estas fotos se ponían directamente en la base de datos. Pero, después de realizar las primeras pruebas con usuarios he realizado que, al añadir un perfil para una persona necesitada en una nueva ciudad no existente en la base de datos, el botón de dicha ciudad queda sin foto y el usuario no la puede modificar. Por eso, he añadido un menú contextual: cuando el usuario mantiene apretada la tarjeta de una ciudad le sale la opción de cambiar la foto.

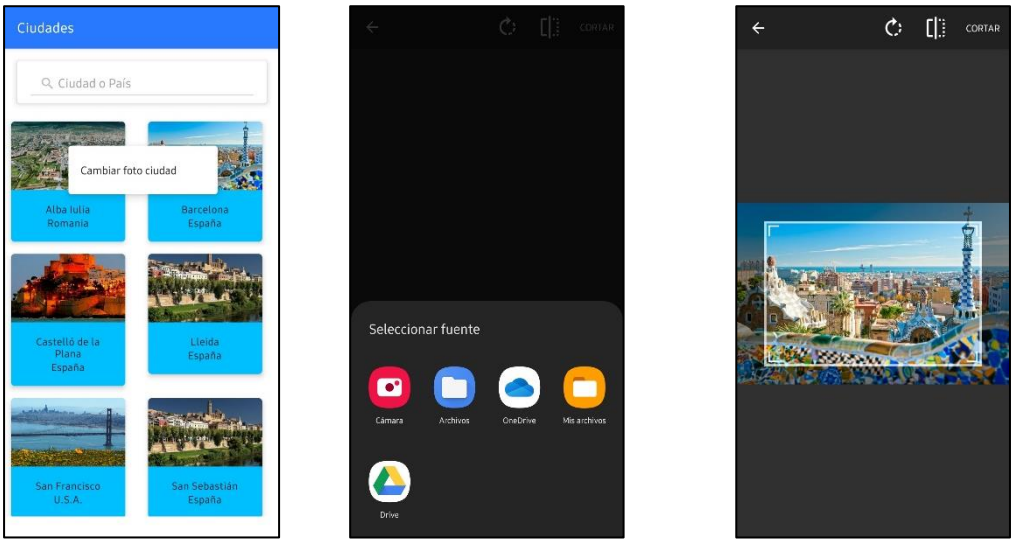


Imagen 69: Utilización librería Android Image Cropper

Para utilizar la librería Android Image Cropper, primero editamos el fichero *build.gradle*, añadiendo la librería: [35]

```
api 'com.theartofdev.edmodo:android-image-cropper:2.8.+'
```

A continuación, tenemos que añadir permisos de lectura y escritura al archivo *AndroidManifest.xml*. Esto permite leer y escribir en el almacenamiento externo.

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

Después, añadimos en el mismo fichero la actividad:

```
<activity android:name="com.theartofdev.edmodo.cropper.CropImageActivity"
  android:theme="@style/Base.Theme.AppCompat"/>
```

Por último, para utilizar la librería, en nuestra actividad añadimos:

```
CropImage.activity()
        .start(getContext(), this);
```

Y sobrescribimos el método *onActivityResult()* que se llama al ejecutarse el código anterior. En este método cambiamos la imagen en la actividad utilizando el Uri[18].

```
@Override
public void onActivityResult(int requestCode, int resultCode, Intent data){

    if (requestCode == CropImage.CROP_IMAGE_ACTIVITY_REQUEST_CODE) {
        CropImage.ActivityResult result = CropImage.getActivityResult(data);
        if (resultCode == RESULT_OK) {
           selectedImagePath = result.getUri();
            homelessProfileImage.setImageURI(selectedImagePath);
        } else if (resultCode ==
CropImage.CROP_IMAGE_ACTIVITY_RESULT_ERROR_CODE) {
            Exception error = result.getError();
        }
    }

}
```

6.2.7. Incluir SwipeRefreshLayout

Hay algunas páginas en las que se deberían ver los cambios al momento: la página principal de donante, la lista de mapas, la página principal del voluntario, la página de entregas y la página de ciudades en el Liquid Galaxy.

Un ejemplo sería editar un perfil para una persona necesitada y cambiar la foto. Al volver a la página principal, la foto no se cambia. Por eso he pensado que estaría bien implementar esta función y que el usuario pueda así actualizar la información al deslizar la página.

Para incluir esta librería, en el archivo *build.gradle* se incluye la dependencia: [67]

```
implementation 'androidx.swiperefreshlayout:swiperefreshlayout:1.0.0'
```

[18] Uri es un identificador de recurso uniforme y puede ser cualquier secuencia sin caracteres especiales

Después, en el *layout* incluimos el elemento que queremos que se actualice dentro del widget SwipeRefreshLayout. En nuestro caso, el *RecyclerView*.

```
<androidx.swiperefreshlayout.widget.SwipeRefreshLayout
    android:id="@+id/swipeRefreshLayout"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="40dp"
    android:layout_marginBottom="@dimen/bottom_margin">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recycler_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:scrollbars="vertical"/>

</androidx.swiperefreshlayout.widget.SwipeRefreshLayout>
```

Por último, en el código del fragmento primero definimos y encontramos el *layout* y después aplicamos *setOnRefreshListner()*, en el que ponemos la acción que se producirá al actualizar la página. En este caso construimos el *RecyclerView* y dejamos la animación durante 3 segundos. También podemos cambiar los colores de la animación utilizando el método *setColorSchemeColors()*.

```
private SwipeRefreshLayout swipeRefreshLayout;
swipeRefreshLayout = view.findViewById(R.id.swipeRefreshLayout);

swipeRefreshLayout.setOnRefreshListener(new
SwipeRefreshLayout.OnRefreshListener() {
    @Override
    public void onRefresh() {
        setUpRecyclerView();
        new Handler().postDelayed(new Runnable() {
            @Override public void run() {
                // Stop animation (This will be after 3 seconds)
                swipeRefreshLayout.setRefreshing(false);
            }
        }, 3000);
    }
});

swipeRefreshLayout.setColorSchemeColors(
        getResources().getColor(R.color.colorPrimary),
        getResources().getColor(R.color.colorAccent),
        getResources().getColor(R.color.green)
);
```

6.2.8. Mostrar información personal en el Liquid Galaxy

Tal como se ha podido observar en los capítulos anteriores, en el Liquid Galaxy aparece información relacionada con las personas necesitadas, los voluntarios y los donantes. Se ha de comentar que en el caso de que la aplicación llegaría a publicarse, en el sistema Liquid Galaxy solamente se mostraría la información relacionada con las personas que necesitan ayuda, ya que son las que han dado su consentimiento para compartir estos datos y, además, los datos de los donantes y los voluntarios no son relevantes. En el caso de los voluntarios y los donantes, quiero mencionar que en ningún momento se recoge su ubicación y la información que se muestra en el Liquid Galaxy no es real y se utiliza para conseguir una mejor experiencia a la hora de realizar la demonstración.

6.2.9. Comprobar información de la red en segundo plano

Como la aplicación necesita utilizar el Internet para funcionar, se ha decidido implementar una clase que compruebe cuando no hay conexión a Internet y avisar al usuario. La aplicación también muestra un aviso cuando la conexión es a través de una red móvil.

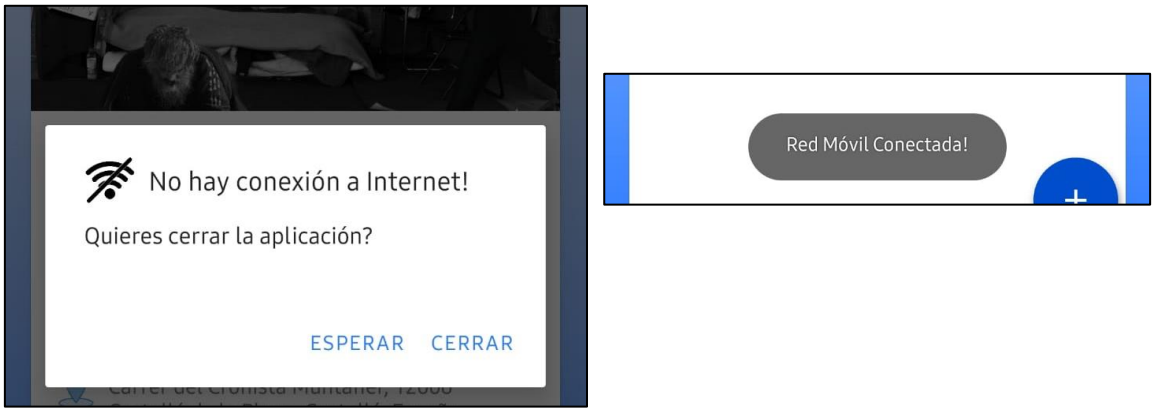


Imagen 70: Gestión red

Para conseguir esto, se ha implementado la actividad *NeworkInfo.java* que se utiliza en todas las actividades principales utilizando el concepto de herencia. [19] Para conseguir que el estado de la red se consulte en segundo plano, se ha utilizado una clase interna que utiliza la clase *AsyncTask*. Esta clase se define por un trabajo que se ejecuta en un hilo secundario y cuyo resultado queremos que se vea en la interfaz principal del usuario. En este caso, se han sobreescrito dos métodos de esta clase: *doInBackground(Parametros ...)* que es donde se realiza la tarea principal y *onPostExecute(Resultado)* que se utiliza para mostrar en la interfaz de usuario el resultado del trabajo.

[19] La herencia es cuando un objeto se basa en otro objeto o clase, usando la misma implementación o comportamiento. Esto es un mecanismo para la reutilización de código para permitirnos extensiones independientes del software original mediante clases públicas e interfaces. [66]

En este caso, en el primer método se ha comprobado si el dispositivo está conectado a *Internet* o no y en el caso que hay conexión, comprobar el tipo (Wi-fi o Móvil).

```
@Override
protected List<String> doInBackground(android.net.NetworkInfo... networkInfos) {
    android.net.NetworkInfo networkInfo = networkInfos[0];
    List<String> stateList = new ArrayList<>();

    if (networkInfo != null && networkInfo.getType() == ConnectivityManager.TYPE_WIFI) {
        stateList.add( getString(R.string.wifi_connected));
    } else if (networkInfo != null && networkInfo.getType() == ConnectivityManager.TYPE_MOBILE) {
        stateList.add(getString(R.string.mobile_connected));
    } else {
        stateList.add(getString(R.string.no_network_operating));
    }

    return stateList;
}
```

Imagen 71: Metodo doInBackground clase CheckNetworkTask()

En el segundo método simplemente comprobamos si es una conexión móvil para mostrar un mensaje emergente y avisar al usuario o en el caso que no hay conexión a Internet, mostrar al usuario un cuadro de dialogo con la opción de esperar o salir de la aplicación.

```
@Override
protected void onPostExecute(List<String> status) {
    super.onPostExecute(status);

    String statusConnection = status.get(0);

    if (statusConnection.equals(getString(R.string.mobile_connected))) {
        Toast.makeText(getApplicationContext(), statusConnection, Toast.LENGTH_SHORT).show();
    }else if (statusConnection.equals(getString(R.string.no_network_operating))) {
        createAlertDialog();
    }
```

Imagen 72: Metodo onPostExecute clase CheckNetworkTask()

También se utiliza el componente *BroadcastRecevier* que permite el registro de eventos en el sistema. De este modo, cuando el evento ocurre, el *Receiver* será notificado por Android. Aquí se utiliza la clase *ConnectivityManager* para poder consultar la red activa y determinar el estado de la red y se ejecuta la tarea en segundo plano.

```java
public class NetworkReceiver extends BroadcastReceiver {

    Context context;

    public NetworkReceiver(Context context){
        this.context = context;
    }
    @Override
    public void onReceive(Context context, Intent intent) {
        ConnectivityManager connectivityManager = (ConnectivityManager) context.getSystemService(Context.CONNECTIVITY_SERVICE);
        android.net.NetworkInfo networkInfo = connectivityManager.getActiveNetworkInfo();

        new CheckNetworkTask(context).execute(networkInfo);
    }
}
```

Imagen 73: Clase NetworkReceiver()

6.3. Desarrollo de la aplicación

A continuación, se comentarán los puntos más destacados que se han utilizado a la hora de desarrollar el proyecto.

6.3.1. Firebase

Tal como se ha ido comentando a lo largo del documento, como base de datos de ha utilizado Firebase. A continuación, se mostrará la estructura de la base de datos.

6.3.1.1. Autenticación

Tal como se ha comentado anteriormente, se utiliza el método de autenticación basado en correo electrónico y contraseña. Utilizando Firebase Authentication, esto es muy fácil de realizar, ya que dispone de un método que permite a los usuarios que se autentifiquen.

Para eso, primero se tiene que añadir la dependencia al fichero Gradle:

```
implementation 'com.google.firebase:firebase-auth:19.3.2'
```

Primero habiliamos el acceso con correo electrónico y contraseña en la consola de Firebase. Despúes, en el método *onCreate()* de la clase *RegisterUserActivity.java* obtenemos la instancia del objeto *FirebaseAuth:*

```
private FirebaseAuth mAuth;
// ...
// Initialize Firebase Auth
mAuth = FirebaseAuth.getInstance();
```

Para crear una cuenta nueva, pasamos la dirección de correo electrónico y la contraseña del usuario nuevo al método *createUserWithEmailAndPassword:*

```
mAuth.createUserWithEmailAndPassword(email, password)
        .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    // Sign in success, update UI with the signed-in user's
information
                    Log.d(TAG, "createUserWithEmail:success");
                    FirebaseUser user = mAuth.getCurrentUser();
                    updateUI(user);
                } else {
                    // If sign in fails, display a message to the user.
                    Log.w(TAG, "createUserWithEmail:failure",
task.getException());
                    Toast.makeText(EmailPasswordActivity.this,
"Authentication failed.",
                            Toast.LENGTH_SHORT).show();
                    updateUI(null);
                }

                // ...
            }
        });
```

En la devolución de llamada, se puede usar el método *getCurrentUser()* para obtener los datos de la cuenta del usuario.

Por último, para que el usuario acceda a su cuenta utilizamos el método *signInWithEmailAndPassword*, comprobando antes si se ha registrad como donante o como voluntario:

```
mAuth.signInWithEmailAndPassword(email, password)
        .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    // Sign in success, update UI with the signed-in user's
information
                    Log.d(TAG, "signInWithEmail:success");
                    FirebaseUser user = mAuth.getCurrentUser();
                    updateUI(user);
                } else {
                    // If sign in fails, display a message to the user.
                    Log.w(TAG, "signInWithEmail:failure",
task.getException());
                    Toast.makeText(EmailPasswordActivity.this,
"Authentication failed.",
                            Toast.LENGTH_SHORT).show();
                    updateUI(null);
                    // ...
```

```
                }

                // ...
            }
        });
```

Finalmente, para salir de la sesión se utiliza el método *signOut* de Firebase:

```
FirebaseAuth.getInstance().signOut();
```

6.3.1.2. Registro usuario

Cuando un usuario entra en la aplicación y quiere registrarse, antes de entrar en el formulario de registro tendrá que escoger que tipo de usuario quiere ser: voluntario o donante. En función del tipo de usuario que elige, tendrá la oportunidad de realizar un tipo de acciones u otras a la hora de iniciar sesión. Para saber el tipo, se utiliza un objeto de tipo *SharedPreferences*, guardando como clave *userType* y como valores *donor* o *volunteer*. De esta manera, la información del formulario de registro se guarda en una colección u otra. Una vez en la pantalla de registro y después de completar todos los campos, el usuario no se puede registrar antes de aceptar los términos y las condiciones de privacidad. Cuando se selecciona un link, tanto en caso de las condiciones legales como en el caso de la política de privacidad se abre una página externa con toda la información. Estas páginas han sido creadas utilizando un generador de términos y condiciones de privacidad [68] y se guardan online utilizando una página web de host para textos. [69]

También se tiene que comentar que, para una mayor seguridad, cuando un usuario se registra en la aplicación, utilizando las funcionalidades de Firebase, se le envía un correo de verificación utilizando el siguiente código: [70]

```
private void sendVerificationEmail(){
    FirebaseAuth auth = FirebaseAuth.getInstance();
    FirebaseUser user = auth.getCurrentUser();

    user.sendEmailVerification()
            .addOnCompleteListener(new OnCompleteListener<Void>() {
                @Override
                public void onComplete(@NonNull Task<Void> task) {
                    if (task.isSuccessful()){
                        showSuccessfulPopup();
                    }
                }
            });
}
```

A la hora de iniciar sesión, se comprueba si el usuario ha verificado su correo electrónico y si no lo ha hecho no podrá iniciar sesión en la aplicación. Para eso se utiliza el método que proporciona Firebase: [70]

```
user = mAuth.getCurrentUser();
if (user.isEmailVerified()){ …//login }
```

Hay que mencionar que esta función es fundamental en el caso de esta aplicación, teniendo en cuenta que es necesario que tanto los donantes como los voluntarios sean reales.

6.3.1.3. Cloud Firestore

Para guardar y gestionar los datos se ha utilizado Cloud Firestore, creado 8 colecciones:

- ***donors:*** se utiliza para guardar la información de los donantes. El nombre de los documentos está formado por el correo con el que se registra el usuario y como campos, aparte de la información personal tenemos dos campos utilizados en las estadísticas que determinan cuantas donaciones a realizado el usuario personalmente y cuantas ha realizado a través de un voluntario. La longitud y la latitud serán utilizadas para poder realizar el vuelo a su posición en las pantallas del Liquid Galaxy.

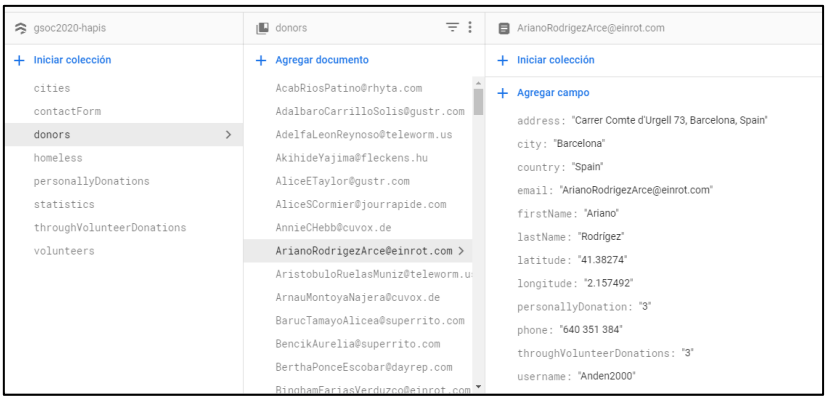


Imagen 74: Colección donors en Firebase

- ***volunteers:*** se utiliza para guardar la información de los usuarios de tipo voluntario. En este caso, para las estadísticas se utiliza un campo que determina cuantos perfiles ha creado para las personas necesitadas.

Imagen 75: Colección volunteers en Firebase

- ***homeless:*** Contiene toda la información relacionada con las personas necesitadas, que se guarda cuando el voluntario crea el perfil. En este caso, también se guarda un link utilizado para poder utilizar la foto de perfil guardada con Cloud Storage.

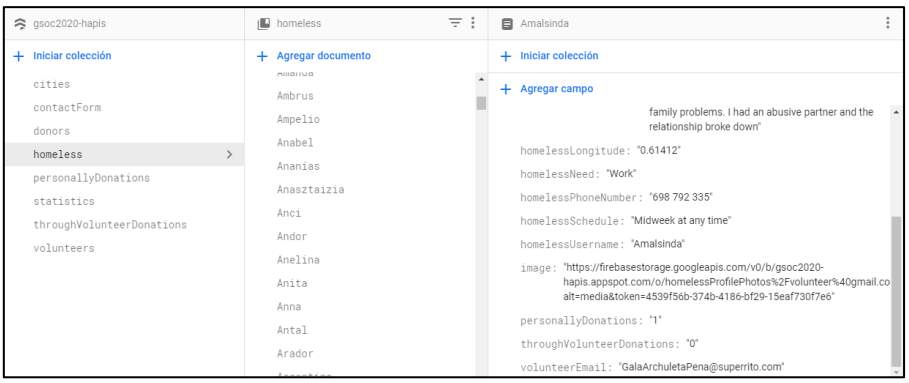


Imagen 76: Colección homeless en Firebase

- ***contactForm:*** Esta colección es necesaria en caso en el que un usuario utiliza el formulario de contacto disponible en la aplicación. Cuando un usuario envía un mensaje en esta colección se guarda un documento con el correo del usuario y el asunto y como campos, el asunto y el mensaje.

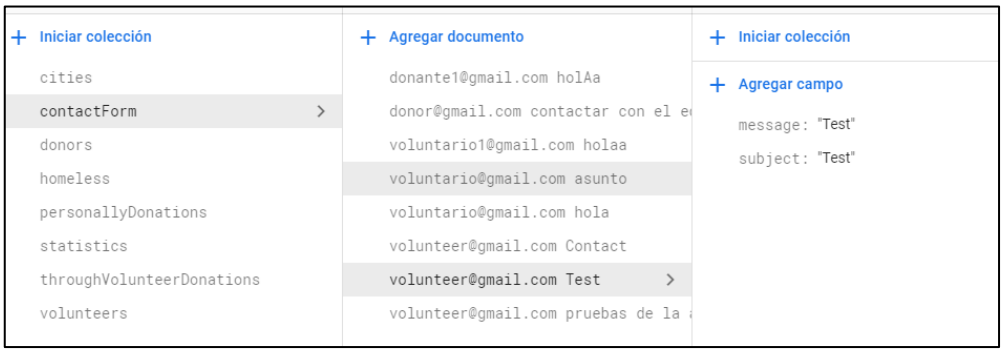


Imagen 77: Colección contactForm en Firebase

- ***personallyDonations:*** En esta colección se guarda información relacionada con las donaciones personales: el correo del donante, el nombre de usuario de la persona necesitada y el tipo de donación. La colección se utiliza únicamente para poder mostrar una ventana con estadísticas en el Liquid Galaxy.

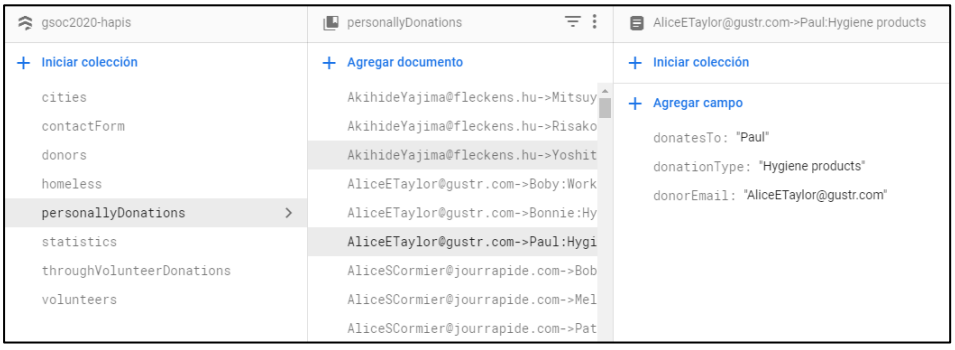


Imagen 78: Colección personallyDonations en Firebase

- ***throughVolunteerDonations:*** Esta colección guarda toda la información relacionada con las entregas de donaciones a través de un voluntario. Los datos de guardan cuando un donante rellena y envía el formulario para realizar una entrega a través de un voluntario. Esta información será utilizada posteriormente en la página *Entrega Donaciones* y será visible para todos los voluntarios. También hay un campo, *deliver* de tipo *boolean* que se actualiza al valor *true* cuando un voluntario pone que ha realizado la entrega. La información de esta colección también será utilizada para mostrar las estadísticas en el Liquid Galaxy.

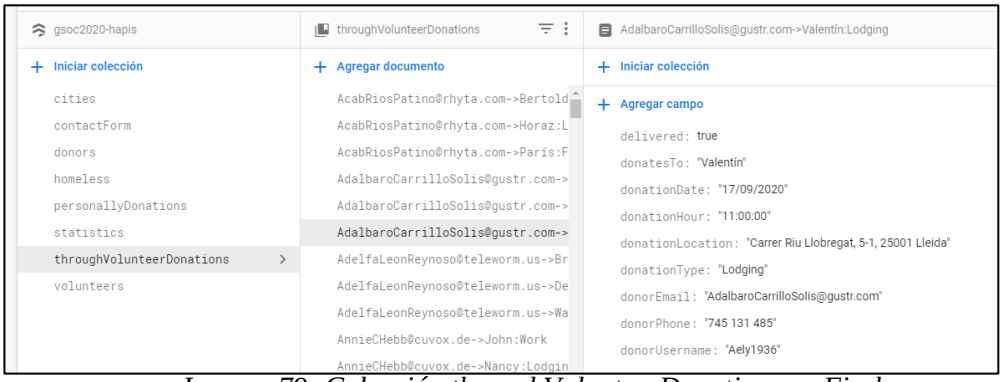


Imagen 79: Colección throughVolunteerDonations en Firebase

- ***statistics:*** Es una colección utilizada únicamente para mostrar una ventana con las estadísticas globales en el Liquid Galaxy. He tenido que crear esta colección porque necesitaba contar todos los documentos de diversas colecciones.

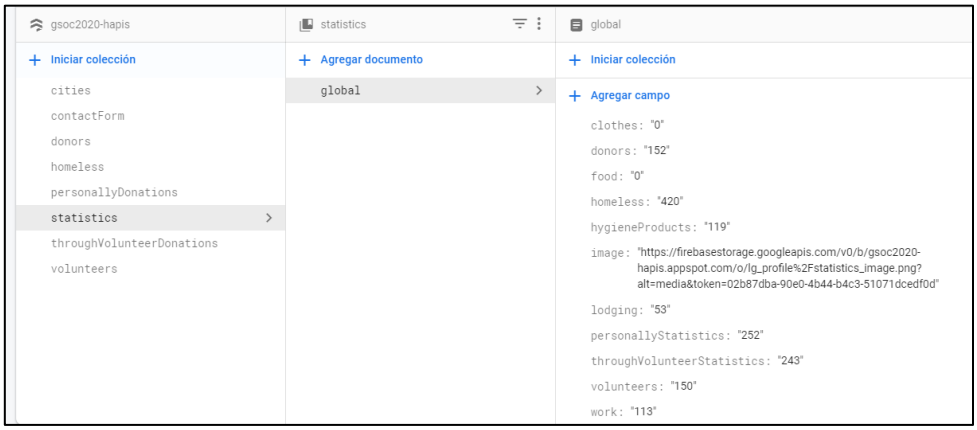


Imagen 80: Colección statistics en Firebase

- ***cities:*** Es una colección utilizada para la parte del Liquid Galaxy que guarda un documento para cada ciudad donde existen personas necesitadas en la base de datos. Se guardan datos relacionados con las estadísticas locales de cada ciudad, el nombre, la altitud y las coordenadas.

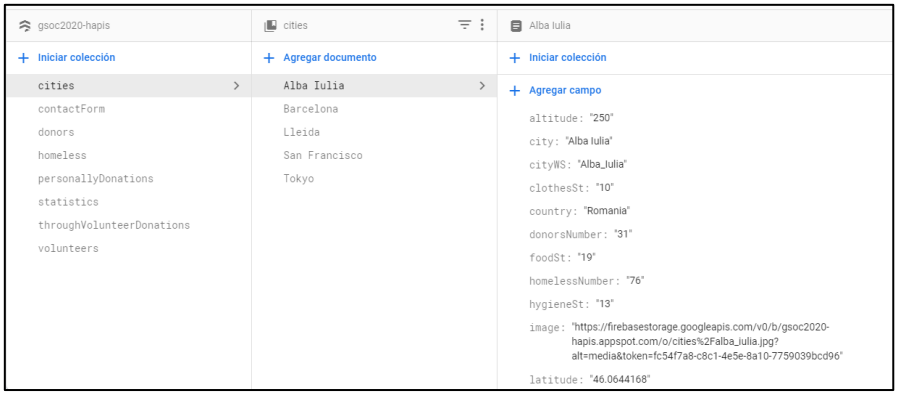


Imagen 81: Colección cities en Firebase

6.3.1.4. Cloud Storage

Cloud Storage se ha utilizado para la siguiente información:

- ***cities:*** guarda una foto de cada ciudad que se mostrará en la ventana de estadísticas locales en el Liquid Galaxy y en las tarjetas de las ciudades en la aplicación.
- ***homelessProfilePhotos:*** en esta colección se encuentran las fotos de perfil de cada persona necesitada.
- ***homelessSignatures:*** aquí se guardan las firmas que realizan las personas necesitadas cuando aceptan los términos y condiciones de privacidad. El nombre de cada documento se guarda con el Nombre y Apellidos de esta persona. Teniendo en cuenta el Reglamento General de protección de datos [71] , antes de que la persona firme, se le mostrarán los avisos de privacidad en un cuadro de texto donde podrá leerlos de una forma simple e inteligible. También se informará sobre el fin con el que se van a utilizar los datos y el hecho de que los datos serán borrados de la base de datos en el momento en el que el voluntario elimine el perfil.

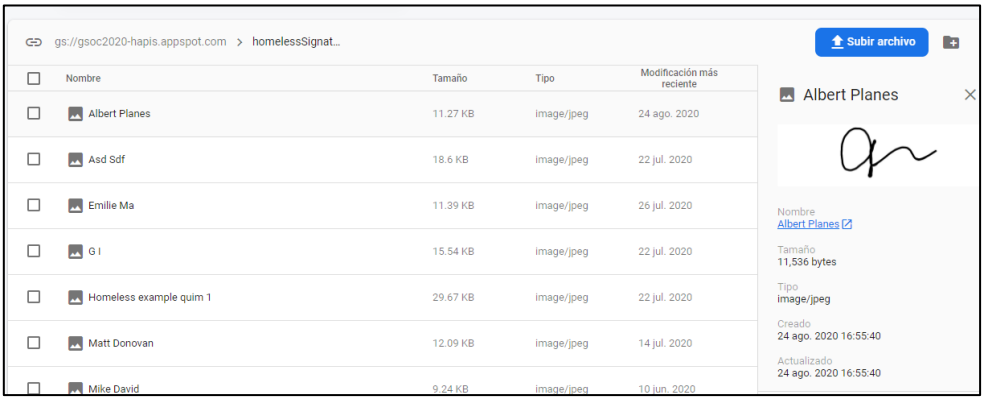


Imagen 82: Colección homelessSignatures en CloudStorage

- ***lg_profile:*** es una colección utilizada para guardar las fotos que se muestran en el Liquid Galaxy en las ventanas de los donantes, los voluntarios, las estadísticas globales y los logos. Estas fotos son fijas y no se pueden subir desde la aplicación. En el caso que se quieren cambiar, se debería hacer desde la consola de Firebase.

6.3.1.5. Firebase Cloud Messaging

También se han utilizado las funciones de Firebase Cloud Messaging para poder enviar una notificación a todos los usuarios con información sobre los nuevos perfiles de personas necesitadas.

Imagen 83: Notificación para todos los usuarios

Para añadir las funciones de Cloud Messaging a la aplicación, es necesario añadir las dependencias en el fichero *build.gradle*: [72]

```
implementation 'com.google.firebase:firebase-messaging:20.2.4'
```

Después de esto, en el archivo *AndroidManifest.xml* se añade un servicio que extienda FirebaseMessagingService. Esto es obligatorio si se desea administrar los mensajes además de recibir notificaciones en apps en segundo plano. Para recibir notificaciones en apps en primer plano, recibir la carga útil de datos y enviar mensajes ascendentes, etc., se tiene que extender este servicio:

```
<service
    android:name=".java.MyFirebaseMessagingService"
    android:exported="false">
    <intent-filter>
```

```
        <action android:name="com.google.firebase.MESSAGING_EVENT" />
    </intent-filter>
</service>
```

Cuando se inicia tu app por primera vez, el SDK de FCM genera un token de registro para la instancia de la app cliente. Si el objetivo son dispositivos individuales o la creación de grupos de dispositivos, es necesario extender FirebaseMessagingService y anular *onNewToken* para acceder a este token.

Después de esto, se tienen que verificar los servicios de Google Play. Una vez configurada la aplicación cliente, se pueden enviar notificaciones desde la consola de Firebase.

En el siguiente gráfico se pueden observar las estádisticas que salen en la consola de Firebase con infomación sobre la cantidad de notificaciones que se le han enviado a los usuarios, cuantas notificaciones han recibido, cuantas han visto y cuantas han abierto. Además lo podemos ver de una manera muy sencilla, pudiendo filtrar esta información según la necesidad.

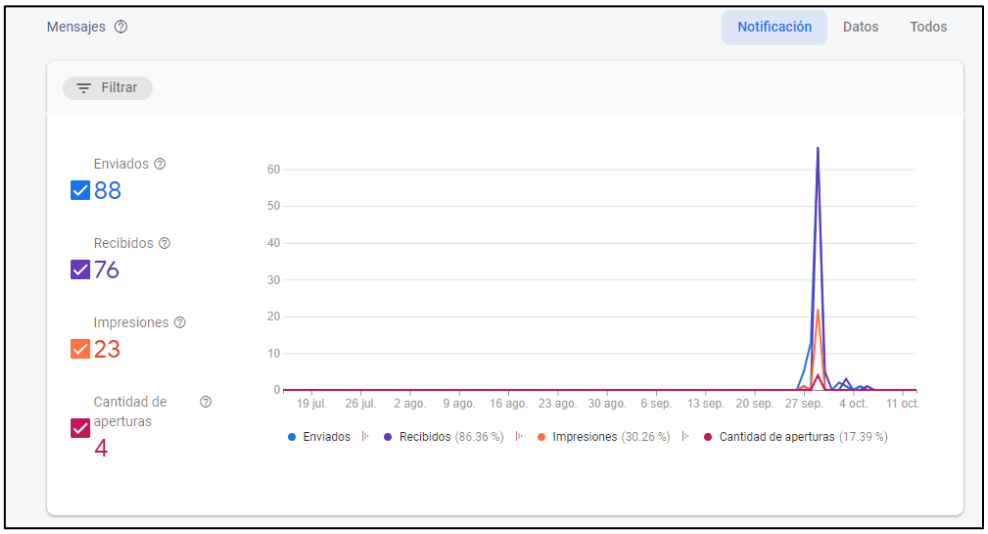


Gráfico 5: Notificaciones Cloud Messaging en la consola Firebase

6.3.1.6. Cloud Functions

Se han incluido las funciones Cloud Functions para enviar una notificación al donante en el momento que el voluntario marca la donación como entregada. [73]

Para utilizar Cloud Functions, necesitamos tener el proyecto agregado en Firebase y por empezar, se tienen que configurar Node.js, que es un entorno utilizado para escribir funciones y Firebase CLI, que es un terminal que proporciona una variedad de herramientas para administrar, implementar y visualizar en proyecto de Firebase.

Después de descargar e instalar el Node.js y npm (node package manager), para instalar la línea de comandos de Firebase a través de npm se utiliza la siguiente comanda:

```
npm install -g firebase-tools
```

Después, se inicializa el SDK de Firebase para Cloud Functions, creando así un proyecto vacío con dependencias y código mínimo de muestra, y se debe elegir TypeScript o JavaScript para redactar funciones.

Para inicializar el proyecto se utilizan los siguientes comandos:

- Se ejecuta *firebase login* para acceder a través del navegador y autenticar la herramienta de Firebase.
- Se va al directorio del proyecto de Firebase.
- Se ejecuta *firebase init functions*. La herramienta ofrece una opción para instalar las dependencias con npm. Es seguro rechazarla si se quiere administrar las dependencias de otra manera, pero, si se hace, habrá que ejecutar *npm install* antes de emular o implementar las funciones.
- La herramienta ofrece dos opciones de lenguaje: JavaScript o TypeScript.

En este caso se ha utilizado JavaScript.

En el archivo *index.js* añadimos el código necesario para gestionar la notificación. Despues de inicializar Firebase, la notificación se envía al actualizarse el campo *delivered* a *true* en la colección *throughVolunteerDonations*. El título y el mensaje son dos campos que se guardan en la base de datos en la colección *donors* y se crean cuando el donante confirma la donación. Por otro lado, el *idToken* también es un campo que se guarda en la información del donante cuando este realiza el inicio de sesión en la aplicación.

```
const functions = require('firebase-functions');
const admin = require('firebase-admin');
admin.initializeApp(functions.config().firebase);
const db = admin.firestore();

exports.sendDeliveredNotification = functions.firestore
    .document('throughVolunteerDonations/{mUid}')
    .onUpdate((change, context) =>{
     const data = change.after.data();
     const title = data.title;
     const content = data.content;
```

```
    if (data.delivered === true){
      var payload = {
        notification: {
          title: title,
          body: content,
        }
      };

      var donorRef = db.collection('donors').doc(data.donorEmail);
        return donorRef
            .get()
            .then(doc => {
              if (!doc.exists) {
                throw new Error('No such User document!');
              } else {
                console.log("Sended")
                admin.messaging().sendToDevice(doc.data().idToken, payload)
                return true
              }
            })
            .catch(err => {
              console.log(err)
              return false;
            });
  }
  return false;
});
```

Imagen 84: Implementación notificaciones push

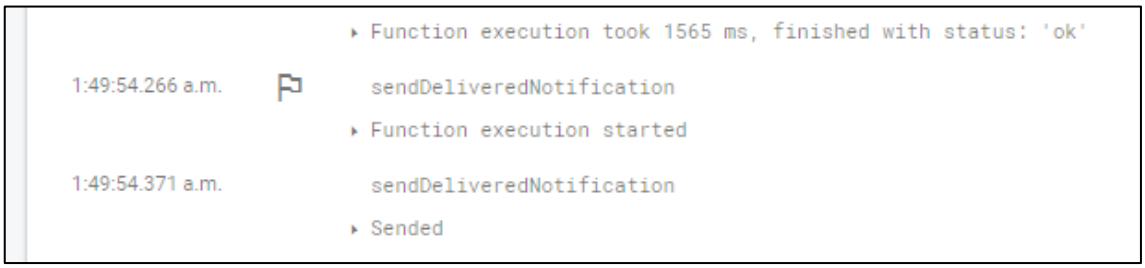


Imagen 85: Registros notificación en la consola de Firebase

Para que la notificación se pueda ver también cuando la aplicación se encuentra en primer plano o cuando no está funcionando, necesitamos implementar el método *onMessageReceived()* en Android. Aquí definimos el icono de la notificación, el título, el mensaje y la acción que se realiza al presionarla. A partir de la versión 7 de Android, también hace falta utilizar un canal. Por tanto, cuando el usuario inicia sesión en la aplicación, se crea también el canal de notificación. [74]

```
int notificationId = new Random().nextInt( bound: 60000);
NotificationCompat.Builder builder = new NotificationCompat.Builder( context: this);
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.N) {
    builder.setSmallIcon(R.mipmap.hapis_icon)
            .setColor(getResources().getColor(R.color.colorPrimary))
            .setContentTitle(remoteMessage.getNotification().getTitle())
            .setContentText(remoteMessage.getNotification().getBody())
            .setStyle(new NotificationCompat.BigTextStyle())
            .setDefaults(Notification.DEFAULT_ALL)
            .setChannelId(getString(R.string.notification_channel))
            .setContentIntent(pendingIntent)
            .setPriority(NotificationManager.IMPORTANCE_HIGH);
} else {
    builder.setSmallIcon(R.mipmap.hapis_icon)
            .setColor(getResources().getColor(R.color.colorPrimary))
            .setContentTitle(remoteMessage.getNotification().getTitle())
            .setContentText(remoteMessage.getNotification().getBody())
            .setDefaults(Notification.DEFAULT_ALL)
            .setStyle(new NotificationCompat.BigTextStyle())
            .setContentIntent(pendingIntent)
            .setPriority(NotificationCompat.PRIORITY_MAX);
}

NotificationManager notificationManager =
        (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);

notificationManager.notify(notificationId, builder.build());
```

```
private void createNotificationChannel() {
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
        CharSequence name = "notification";
        String description = "notificationD";
        int importance = NotificationManager.IMPORTANCE_HIGH;
        NotificationChannel channel = new NotificationChannel("notification", name, importance);
        channel.setDescription(description);
        NotificationManager notificationManager = getSystemService(NotificationManager.class);
        notificationManager.createNotificationChannel(channel);
    }
}
```

Imagen 86: Implementación método onMessageReceive y creación canal

6.3.2. Funciones Liquid Galaxy

A continuación, se describirán las funciones utilizadas para mostrar la información en el Liquid Galaxy.

6.3.2.1. Instalación

Antes de empezar, para la instalación de un sistema Liquid Galaxy se necesitan mínimo 3 máquina que tengan instalado el sistema operativo Ubuntu 16.04, porque de otro modo el Liquid Galaxy no va a funcionar. [75]

A la hora de instalar el sistema operativo en cada una de las maquinas, se configura como nombre de usuario: *lg* y como nombre de la maquina: *lgX*, siendo la *X* el número de la pantalla y *lg1* el master. Luego, las maquinas se ponen en el sentido de las agujas del reloj.

Antes de empezar la instalación, se tienen que instalar algunos paquetes y luego actualizar la base de datos de paquetes y el sistema en cada una de las maquinas, utilizando los comandos siguientes:

```
$ sudo apt install lsb-core

$ sudo apt install lsb

$ sudo apt update

$ sudo apt upgrade -f
```

A continuación, se copia la siguiente comanda para obtener y ejecutar el archivo de instalación:

```
$bash <(curl -s https://raw.githubusercontent.com/LiquidGalaxyLAB/liquid-galaxy/master/install.sh)
```

Cuando se ejecuta el archivo, se pedirá la siguiente información:

En el master:

- ***Machide ID:*** el número de la maquina
- ***Total machines count:*** el número total de maquinas
- ***LG Frames:*** el orden de las maquinas
- ***Unique number:*** un número entero único que identifica la instalación, para evitar conflicto con otros sistemas Liquid Galaxy existentes en la misma red

En los slaves:

- ***Master IP:*** La IP del master
- ***Master local password:*** la contraseña del master (casi siempre se utiliza *lg* como contraseña en todas las maquinas)

6.3.2.2. Conexión

La conexión de la aplicación Android con el sistema Liquid Galaxy se realiza a través del protocolo SSH. [76]

En la implementación se utiliza una librería llamada JSCH que es una librería escrita en Java y que permite la conexión a un servidor SSH y usar el reenvío de puertos, la transferencia de archivos etc.

En nuestro caso, se ha creado una clase que implementa la interfaz *Runnable* por tal de que se ejecute en un hilo diferente en la cual se utilizan el nombre de usuario, contraseña, IP y puerto para realizar la conexión.

En esta clase también tenemos un método para enviar los comandos al master del Liquid Galaxy.

7. Conclusiones, trabajo futuro y posibles mejoras

Tal como se puede observar a lo largo de este documento, este proyecto en un principio solamente era una idea para realizar una aplicación a través de la cual se pueden ayudar a los más necesitados. Pero, a la hora de diseñar y sobretodo implementar la aplicación me he dado cuenta de muchas cosas que no se pueden realizar y he tenido que cambiar todo varias veces. También he tenido que buscar soluciones para que la aplicación se pueda realizar sin ayuda de organizaciones específicas y de este modo conseguir que las personas que quieren ayudar tengan la oportunidad de conocer en persona a los que ayudan.

Quiero mencionar que llegar aquí ha sido un largo camino, en el que he aprendido mucho sobre las diversas tecnologías que he utilizado y aun así estoy segura que todavía queda mucho por mejorar.

En cuanto al trabajo futuro, me gustaría mucho poder publicar de verdad esta aplicación y conseguir ayudar a muchas personas necesitadas, sobretodo porque a través de las presentaciones realizadas para el programa Google Summer Of Code, he conseguido dar bastante visibilidad al proyecto y creo que las personas estarían interesadas en una aplicación como esta.

En este momento, estoy colaborando con los laboratorios TIC del Parque Científico de Lleida y el grupo Women Techmaker Lleida [77], del que formo parte para conseguir que este proyecto salga de verdad.

De momento, hemos creado las redes sociales y una página web del proyecto [78] para obtener más visibilidad.

También quiero comentar que hemos tenido una reunión con el abogado Ramon Arnó para pedirle algunos consejos sobre los aspectos legales y nos ha ayudado muchísimo para saber cómo actuar a la hora de publicar la aplicación.

Primero, se debería registrar la marca para asegurarnos que ningún otra persona la utiliza o la va a utilizar en el futuro. Después, habría que informar al usuario donde se aloja la base de datos y en este caso debería alojarse en la Unión Europea.

En cuanto a la privacidad habría que hacer algunos cambios en la aplicación para conseguir cumplir con dos conceptos muy importantes: privacidad por defecto y privacidad por diseño. Habría que conseguir que los usuarios mantengan sus datos totalmente anónimos si así lo desean.

En cuanto a los voluntarios, habría que conseguir una colaboración con alguna entidad, ya que conforme la Ley del Voluntariado [79] , se necesita un seguro por cada voluntario. Además, considerando toda la información comentada anteriormente cuando un voluntario se registra, antes de poder iniciar sesión en la aplicación habría que realizar un proceso de admisión por videoconferencia o personal si es posible. De este modo evitaríamos el fraude y conseguiríamos más seguridad.

Otro cambio que debería realizarse sería cambiar la opción de firma en caso de las condiciones de uso cuando se crea un perfil para una persona necesitada y añadir la opción de que el voluntario haga una foto a la persona necesitada o buscar algún otro método de confirmación. Esto se ha decidido porque así el voluntario no tendría que entregar su teléfono móvil a la persona necesitada para firmar y debido a la situación actual, sería lo mejor.

Como funcionalidades, me gustaría poder incluir en un futuro un chat entre el donante y el voluntario que ha creado el perfil de la persona necesitada. De este modo la comunicación sería mucho más rápida y además, el donante ya no tendría que facilitar su correo electrónico o teléfono si no lo desea.

Pero, pese a todos estos cambios que se deberían realizar tenemos pensado publicar la aplicación en la primavera del año 2021.

Por último, quiero comentar que estoy muy orgullosa de haber podido crear y desarrollar este proyecto y tengo muchas ganas de conseguir publicar la aplicación y ayudar a muchas personas.

Bibliografía

[1] «About | Google Summer of Code». https://summerofcode.withgoogle.com/about/ (accedido oct. 04, 2020).

[2] «Estilo IEEE - Citas y elaboración de bibliografía: el plagio y el uso ético de la información - Biblioguías at Universidad Autónoma de Madrid». https://biblioguias.uam.es/citar/estilo_ieee (accedido oct. 05, 2020).

[3] «Liquid Galaxy project community site». https://www.liquidgalaxy.eu/ (accedido oct. 05, 2020).

[4] «Informe AROPE sobre el Estado de la Pobreza en España». https://www.eapn.es/estadodepobreza/ (accedido oct. 05, 2020).

[5] «EL ESTADO DE LA POBREZA SEGUIMIENTO DEL INDICADOR DE POBREZA Y EXCLUSIÓN SOCIAL EN ESPAÑA», 2008. Accedido: oct. 05, 2020. [En línea]. Disponible en: www.eapn.es.

[6] «Informe Pobreza y Desigualdad 230620 EMB.pdf | Con la tecnología de Box». https://oxfam.app.box.com/s/90ijkmfp0tlfzd1q7hb64mka4q29xfvt (accedido oct. 05, 2020).

[7] «3.5.1 Tipos de riesgos - Gestion de Proyectos Software». https://sites.google.com/site/gestiondeproyectossoftware/unidad-3-planificacion-de-proyecto/3-5-1-tipos-de-riesgos (accedido oct. 05, 2020).

[8] «Trello». https://trello.com/ (accedido oct. 05, 2020).

[9] «¿Qué es el análisis de la competencia y cómo hacer uno? | CreceNegocios». https://www.crecenegocios.com/analisis-de-la-competencia/ (accedido oct. 05, 2020).

[10] «ShareTheMeal: Dona a una buena causa - Aplicaciones en Google Play». https://play.google.com/store/apps/details?id=org.sharethemeal.app (accedido oct. 05, 2020).

[11] «Charity Miles: Walking & Running Distance Tracker - Apps on Google Play». https://play.google.com/store/apps/details?id=com.charitymilescm.android&hl=en (accedido oct. 05, 2020).

[12] «OLIO - Comparte más. Desperdicia menos. - Aplicaciones en Google Play». https://play.google.com/store/apps/details?hl=es&id=com.olioex.android&rdid=com.olioex.android (accedido oct. 05, 2020).

[13] «OurCalling - Aplicaciones en Google Play». https://play.google.com/store/apps/details?id=com.ourcalling.homeless&hl=es (accedido oct. 05, 2020).

[14] «Samaritan - Aplicaciones en Google Play». https://play.google.com/store/apps/details?id=org.givesafe.app (accedido oct. 05, 2020).

[15] «DAFO: Herramienta de planificación estratégica | Pascual Parada - Consultor de estrategia digital y de crecimiento. Mentor y formador para empresas y Startups.» http://www.pascualparada.com/dafo-herramienta-de-planificacion-estrategica/

(accedido oct. 05, 2020).

[16] «Modelo de Proceso de la Ingeniería de la Usabilidad y la Accesibilidad (MPIu+a) | Curso de Interacción Persona-Ordenador». https://mpiua.invid.udl.cat/fases-mpiua/ (accedido oct. 05, 2020).

[17] «Pre Evaluation Form Hapis». https://docs.google.com/forms/d/e/1FAIpQLSdH16qbcJBOWVDYXEy9taK1C9Mr_pnH3c_IXFe9LwQFKExs5w/viewform (accedido oct. 15, 2020).

[18] «Online User Testing Tool | Loop11». https://www.loop11.com/ (accedido oct. 15, 2020).

[19] «Sin acceso | IAB Spain». https://iabspain.es/sin-acceso/download-id/24646/ (accedido oct. 05, 2020).

[20] «Android supera el 90% de cuota en España mientras que iOS cae por debajo del 9%, según Kantar». https://www.xatakamovil.com/mercado/android-supera-90-cuota-espana-ios-cae-debajo-9-kantar (accedido oct. 05, 2020).

[21] «Arquitectura de la plataforma | Desarrolladores de Android». https://developer.android.com/guide/platform (accedido oct. 05, 2020).

[22] «Liquid Galaxy». https://liquidgalaxy.org/ (accedido oct. 05, 2020).

[23] «¿Qué Es El Protocolo SSH Y Cómo Funciona?» https://www.hostinger.es/tutoriales/que-es-ssh (accedido oct. 05, 2020).

[24] «Tutorial de KML | Keyhole Markup Language | Google Developers». https://developers.google.com/kml/documentation/kml_tut#marcas-de-posición (accedido oct. 05, 2020).

[25] «Google Earth». https://www.google.com/intl/es_ALL/earth/ (accedido oct. 05, 2020).

[26] «GitHub - Wikipedia, la enciclopedia libre». https://es.wikipedia.org/wiki/GitHub (accedido oct. 05, 2020).

[27] «Qué son los Servicios de Google Play y para qué sirven». https://www.xatakandroid.com/sistema-operativo/que-servicios-google-play-sirven (accedido oct. 05, 2020).

[28] «braintree/android-card-form: A ready-made card form layout that can be included in your Android app, making it easy to accept credit and debit cards.» https://github.com/braintree/android-card-form (accedido jul. 19, 2020).

[29] «AbhinayMe/currency-edittext: A Custom EditText implementation that allows formatting of currency-based numeric inputs.» https://github.com/AbhinayMe/currency-edittext (accedido oct. 05, 2020).

[30] «Clans/FloatingActionButton: Android Floating Action Button based on Material Design specification». https://github.com/Clans/FloatingActionButton (accedido oct. 05, 2020).

[31] «szimek/signature_pad: HTML5 canvas based smooth signature drawing». https://github.com/szimek/signature_pad (accedido oct. 05, 2020).

[32] «bumptech/glide: An image loading and caching library for Android focused on smooth scrolling». https://github.com/bumptech/glide (accedido oct. 05, 2020).

[33] «YouTube Android Player API | YouTube API de Android».

https://developers.google.com/youtube/android/player (accedido oct. 05, 2020).

[34] «JSch - Java Secure Channel». http://www.jcraft.com/jsch/ (accedido sep. 08, 2020).

[35] «ArthurHub/Android-Image-Cropper: Image Cropping Library for Android, optimized for Camera / Gallery.» https://github.com/ArthurHub/Android-Image-Cropper (accedido oct. 05, 2020).

[36] «Principales Características de Java». http://personales.upv.es/rmartin/cursoJava/Java/Introduccion/PrincipalesCaracteristicas.htm (accedido oct. 05, 2020).

[37] «¿Qué es Kotlin y para qué se puede utilizar? | Quality Devs». https://www.qualitydevs.com/2019/10/02/que-es-kotlin/ (accedido oct. 05, 2020).

[38] «Introducción a Android Studio | Desarrolladores de Android». https://developer.android.com/studio/intro?hl=es-419 (accedido sep. 02, 2020).

[39] «Diagramas de Casos de Uso | LENGUAJE DE MODELADO UNIFICADO UML». http://stadium.unad.edu.co/ovas/10596_9839/diagramas_de_casos_de_uso.html (accedido oct. 05, 2020).

[40] «diagrams.net». https://app.diagrams.net/ (accedido oct. 05, 2020).

[41] «Download Android Studio and SDK tools | Android Studio». https://developer.android.com/studio (accedido oct. 05, 2020).

[42] «Ubuntu 16.04.7 LTS (Xenial Xerus)». https://releases.ubuntu.com/16.04.7/?_ga=2.153551357.418405133.1599677304-1224323010.1599677304 (accedido oct. 05, 2020).

[43] «liquid-galaxy/google-earth-pro7_1.deb at master · LiquidGalaxyLAB/liquid-galaxy». https://github.com/LiquidGalaxyLAB/liquid-galaxy/blob/master/google-earth-pro7_1.deb (accedido oct. 05, 2020).

[44] «Modelo relacional - Wikipedia, la enciclopedia libre». https://es.wikipedia.org/wiki/Modelo_relacional (accedido oct. 05, 2020).

[45] «Free prototyping tool for web & mobile apps - Justinmind». https://www.justinmind.com/ (accedido oct. 05, 2020).

[46] «Herramienta de diseño y colaboración de experiencias e interfaces de usuario | Adobe XD». https://www.adobe.com/es/products/xd.html (accedido oct. 06, 2020).

[47] «Crear, insertar y editar dibujos - Ordenador - Ayuda de Editores de Documentos». https://support.google.com/docs/answer/179740?co=GENIE.Platform%3DDesktop&hl=es (accedido oct. 06, 2020).

[48] «Introducción a las actividades | Desarrolladores de Android». https://developer.android.com/guide/components/activities/intro-activities?hl=es (accedido oct. 06, 2020).

[49] «Fragmentos | Desarrolladores de Android | Android Developers». https://developer.android.com/guide/components/fragments?hl=es-419 (accedido oct. 06, 2020).

[50] «Material Design - Wikipedia, la enciclopedia libre». https://es.wikipedia.org/wiki/Material_Design (accedido oct. 06, 2020).

[51] «Navigation drawer - Material Design». https://material.io/components/navigation-drawer (accedido oct. 06, 2020).

[52] «Cómo crear una lista con RecyclerView | Desarrolladores de Android». https://developer.android.com/guide/topics/ui/layout/recyclerview?hl=es-419 (accedido oct. 05, 2020).

[53] «Cómo deslizarse entre fragmentos con ViewPager». https://developer.android.com/training/animation/screen-slide?hl=es (accedido oct. 06, 2020).

[54] «ChipGroup | Desarrolladores de Android | Android Developers». https://developer.android.com/reference/com/google/android/material/chip/ChipGroup (accedido oct. 06, 2020).

[55] «Bottom navigation - Material Design». https://material.io/develop/android/components/bottom-navigation (accedido oct. 06, 2020).

[56] «Modo lite | Maps SDK for Android | Google Developers». https://developers.google.com/maps/documentation/android-sdk/lite (accedido oct. 06, 2020).

[57] «Diferencias entre la carpeta drawable y mimap en Android - Diego Laballós». https://diegolaballos.com/blog/diferencias-entre-la-carpeta-drawable-y-mimap-en-android/ (accedido oct. 06, 2020).

[58] «Android Asset Studio». https://romannurik.github.io/AndroidAssetStudio/index.html (accedido sep. 02, 2020).

[59] «¿Como implementar splash screen correctamente en Android? | by Sneyder Angulo | Medium». https://medium.com/@sneyderangulo/como-implementar-splash-screen-correctamente-en-android-f6abcc592b4c (accedido sep. 02, 2020).

[60] «Blue UI Icons - Download Free Icons PNG and SVG | Icons8». https://icons8.com/icons/ultraviolet (accedido sep. 01, 2020).

[61] «CleanPNG - HD png images and illustrations. Free unlimited download. - CleanPNG / KissPNG». https://www.cleanpng.com/ (accedido sep. 01, 2020).

[62] «Beautiful Free Images & Pictures | Unsplash». https://unsplash.com/ (accedido sep. 01, 2020).

[63] «Homeless fotos database - Documentos de Google». https://docs.google.com/document/d/1ZjpoeqWUgFagAW3VPulsK9YFz4c9bfRJnHt1xYwzjc4/edit (accedido sep. 01, 2020).

[64] «ClaudiaMihaelaDiosan/Hapis_MihaelaClaudiaDiosan». https://github.com/ClaudiaMihaelaDiosan/Hapis_MihaelaClaudiaDiosan (accedido oct. 15, 2020).

[65] «Descripción general del manifiesto de una app». https://developer.android.com/guide/topics/manifest/manifest-intro?hl=es-419#perms (accedido sep. 08, 2020).

[66] «Herencia vs Composición ¿Tienes claro cuál es el rival más débil?» https://devexperto.com/herencia-vs-composicion/ (accedido oct. 13, 2020).

[67] «Swiperefreshlayout | Desarrolladores de Android | Android Developers». https://developer.android.com/jetpack/androidx/releases/swiperefreshlayout (accedido oct. 06, 2020).

[68] «Free Terms and Conditions Generator for EU Businesses». https://termly.io/en/products/terms-and-conditions-generator/ (accedido sep. 08, 2020).

[69] «Free Text Host - Free Easy Text Hosting». https://freetexthost.net/ (accedido sep. 08, 2020).

[70] «Administra usuarios en Firebase». https://firebase.google.com/docs/auth/android/manage-users?hl=es (accedido oct. 06, 2020).

[71] «RGPD - Reglamento General de Protección de datos». https://rgpd.es/ (accedido sep. 08, 2020).

[72] «Configura una app cliente de Firebase Cloud Messaging en Android». https://firebase.google.com/docs/cloud-messaging/android/client?hl=es (accedido oct. 06, 2020).

[73] «Primeros pasos: Escribe, prueba e implementa tus primeras funciones». https://firebase.google.com/docs/functions/get-started?hl=es (accedido sep. 10, 2020).

[74] «Recibe mensajes en una app para Android | Firebase». https://firebase.google.com/docs/cloud-messaging/android/receive (accedido oct. 06, 2020).

[75] «LiquidGalaxyLAB/liquid-galaxy: Liquid Galaxy core.» https://github.com/LiquidGalaxyLAB/liquid-galaxy (accedido sep. 08, 2020).

[76] «Secure Shell - Wikipedia, la enciclopedia libre». https://es.wikipedia.org/wiki/Secure_Shell (accedido sep. 08, 2020).

[77] «WomenTechMakers Lleida». https://www.womentechmakerslleida.com/ (accedido sep. 08, 2020).

[78] «Proyecto HAPIS». https://www.proyectohapis.com/ (accedido oct. 06, 2020).

[79] «(No Title)». https://www.boe.es/buscar/pdf/2015/BOE-A-2015-9726-consolidado.pdf (accedido oct. 06, 2020).

[80] «¿Qué es un Smartphone? Conoce todos los detalles | WhistleOut». https://www.whistleout.com.mx/CellPhones/Guides/que-es-un-smartphone (accedido oct. 05, 2020).

[81] «Qué es Android | Android». https://www.android.com/intl/es_es/what-is-android/ (accedido oct. 05, 2020).

[82] «Productos y Servicios / Publicaciones / Publicaciones de descarga gratuita». https://www.ine.es/ss/Satellite?L=es_ES&c=INESeccion_C&cid=1259941637944&p=1254735110672&pagename=ProductosYServicios/PYSLayout (accedido oct. 05, 2020).

[83] «8 cosas que deberías saber sobre los beacons - Accent Systems». https://accent-systems.com/es/blog/8-cosas-que-deberias-saber-sobre-los-beacons/ (accedido oct. 05, 2020).

[84] «iOS - Wikipedia, la enciclopedia libre». https://es.wikipedia.org/wiki/IOS (accedido oct.

05, 2020).

[85] «Interfaz de usuario en Android: Layouts | sgoliver.net». https://www.sgoliver.net/blog/interfaz-de-usuario-en-android-layouts/ (accedido oct. 06, 2020).

Anexos

Anexo 1: Otros casos de uso

Caso de uso 4: Iniciar sesión

Actores: usuario (donante o voluntario)

Precondición: el usuario ya está registrado en el sistema

Postcondición: el usuario inicia sesión en el sistema

Actor	Sistema	Backend
1. El usuario quiere iniciar sesión en la aplicación. Introduce el correo electrónico y la contraseña.		
	2. El sistema recibe los datos y los envía al backend para validar que los credenciales están correctos.	
		3. El backend procesa los datos y envía respuesta al sistema.
	6. El sistema muestra la pantalla inicial con los perfiles de la personas necesitadas ya existentes.	
7. El usuario ha iniciado sesión en el sistema.		

Tabla 13: Caso de uso 4: Iniciar sesión

Cursos alternativos:

2a, 3a, 4a – No hay conexión a Internet:

El sistema muestra un mensaje en la pantalla conforma no hay conexión a Internet.

3b – El correo electrónico o la contraseña son incorrectas:

El sistema devuelve un mensaje de error.

Requisititos especiales:

Para iniciar sesión en el sistema hace falta una conexión a Internet.

Caso de uso 5: Cerrar sesión

Actores: usuario (donante o voluntario)

Precondición: el usuario tiene sesión iniciada en el sistema

Postcondición: el usuario se ha desconectado del sistema

Actor	Sistema	Backend
1. El usuario quiere cerrar sesión en la aplicación.		
	2. El sistema recibe los datos y envía la petición de desconexión al backend.	
		3. El backend desconecta el usuario.
	6. El sistema muestra la pantalla inicial de login.	
7. El usuario ha cerrado sesión en el sistema.		

Tabla 14: Caso de uso 5: Cerrar sesión

Cursos alternativos:

No hay conexión a Internet:

El sistema muestra un mensaje en la pantalla conforme no hay conexión a Internet.

Requisititos especiales:

Para cerrar sesión correctamente en el sistema hace falta una conexión a Internet.

Caso de uso 6: Realizar donación

Actores: usuario (donante o voluntario)

Precondición: el usuario tiene sesión iniciada en el sistema

Postcondición: el usuario ha realizado una donación

Actor	Sistema	Backend
1. El usuario quiere realizar una donación.		
	2. El sistema muestra una pantalla en la que el usuario puede escoger si realiza una donación de dinero o mira un video patrocinado para realizar la donación.	
3. El usuario escoge realizar la donación con tarjeta de crédito o débito.		
	4. El sistema muestra la pantalla con un formulario para rellenar los datos de pago.	
		5. El servidor configurado anteriormente procesa el pago.
	6. El sistema muestra un mensaje conforme la donación se ha realizado correctamente.	
7. El usuario ha realizado correctamente la donación.		

Tabla 15: Caso de uso 6: Realizar donación

Cursos alternativos:

2a, 4a, 5a – No hay conexión a Internet:

El sistema muestra un mensaje en la pantalla conforme no hay conexión a Internet.

2b – El usuario escoge mirar un video patrocinado:

El sistema muestra un video de no más de 30 segundos. El usuario mira el video y el patrocinador realiza una donación para apoyar los desarrolladores de la aplicación.

3a – El usuario introduce datos de pago incorrectos:

El sistema muestra un mensaje de error con el texto necesario.

Requisititos especiales:

Para realizar el pago o mirar el video en el sistema hace falta una conexión a Internet.

Caso de uso 7: Contactar

Actores: usuario (donante o voluntario)

Precondición: el usuario tiene sesión iniciada en el sistema

Postcondición: el usuario ha contactado con el equipo de la aplicación

Actor	Sistema	Backend
1. El usuario quiere contactar con el equipo de la aplicación.		
	2. El sistema muestra una pantalla con un formulario para que el usuario rellene el asunto y el mensaje.	
		5. El backend guarda la información en la base de datos.
	6. El sistema muestra un mensaje conforme el mensaje se ha enviado correctamente.	
7. El usuario ha enviado el mensaje correctamente.		

Tabla 16: Caso de uso 7: Contactar

Cursos alternativos:

2a, 5a, 6a – No hay conexión a Internet:

El sistema muestra un mensaje en la pantalla conforme no hay conexión a Internet.

2b – El usuario intenta contactar con el asunto o el mensaje vacío:

El sistema muestra un mensaje de error conforme los dos campos son obligatorios.

Requisititos especiales:

Para contactar con el equipo de la aplicación, el sistema hace falta una conexión a Internet.

Caso de uso 8: Gestionar entrega de donaciones

Actores: usuario (voluntario, donante)

Precondición: el usuario tiene sesión iniciada en el sistema

Postcondición: el usuario ha gestionado la entrega de donación

Actor	Sistema	Backend
1. El usuario registrado como voluntario quiere gestionar la entrega de donaciones		
		2. El backend proporciona los datos guardados en la base de datos.
	3. El sistema muestra una pantalla con un listado en el que aparece la información necesaria en caso que el donante quiere realizar la donación a través de un voluntario.	
4. El usuario selecciona la entrega ya realizada.		
	5. El sistema muestra un dialogo de confirmación para comprobar que la entrega ya se ha realizado	
6. El usuario confirma la entrega.		
		7. El backend gestiona los datos a través de un campo de tipo boolean.
	8. El sistema muestra un mensaje para dar las gracias al voluntario para su colaboración.	
		9. El backend envía una notificación push al donante para confirmar la entrega.
10. El usuario registrado como donante recibe la notificación push.		

11. El voluntario ha gestionado la entrega.		

Tabla 17: Caso de uso 8: Gestionar entrega de donaciones

Cursos alternativos:

2a, 3a, 7a, 9a – No hay conexión a Internet:

El sistema muestra un mensaje en la pantalla conforme no hay conexión a Internet.

Requisititos especiales:

Para gestionar la entrega de donaciones el sistema hace falta una conexión a Internet.

Caso de uso 9: Visualizar datos en el mapa

Actores: usuario registrado como donante

Precondición: el usuario tiene sesión iniciada en el sistema

Postcondición: el usuario ha visualizado correctamente los datos en el mapa

Actor	Sistema	Backend
1. El usuario registrado como donante quiere visualizar las personas necesitadas en el mapa.		
		2. El backend proporciona los datos guardados en la base de datos.
	3. El sistema muestra un mapa con marcadores para cada uno de los perfiles de persona necesitada registrados en la aplicación	
4. El usuario selecciona un marcador en el mapa.		
	5. El sistema muestra un cuadro con información sobre la persona seleccionada.	

11. El donante ha visualizado los datos correctamente.		

Tabla 18: Caso de uso 9: Visualizar datos en el mapa

Cursos alternativos:

2a, 3a, 5a – No hay conexión a Internet:

El sistema muestra un mensaje en la pantalla conforme no hay conexión a Internet.

3b – El usuario ha seleccionado anteriormente que solo quiere visualizar el mapa con Wifi y no hay conexión Wifi en el dispositivo:

El sistema muestra un snackbar para que el usuario pueda activar la Wifi.

Si no hay conexión Wifi disponible, el usuario puede cambiar la vista del mapa utilizando cualquier red.

Requisititos especiales:

Para visualizar correctamente los datos en el mapa hace falta una conexión a Internet.

Caso de uso 10: Visualizar lista de mapas

Actores: usuario registrado como donante

Precondición: el usuario tiene sesión iniciada en el sistema

Postcondición: el usuario ha visualizado correctamente la lista de mapas

Actor	Sistema	Backend
1. El usuario registrado como donante quiere visualizar una lista de mapas con información sobre las personas necesitads.		

		2. El backend proporciona los datos guardados en la base de datos.
	3. El sistema muestra una lista mapas con la ubicación, el nombre de usuario y la dirección de cada persona necesitada registrada en el sistema.	
4. El usuario selecciona un el botón de ruta en el mapa.		
	5. El sistema abre la aplicación Google Maps y muestra la ruta para llegar a la ubicación de la persona necesitada.	
11. El donante ha visualizado los datos correctamente.		

Tabla 19: Caso de uso 10: Visualizar lista de mapas

Cursos alternativos:

2a, 3a, 5a – No hay conexión a Internet:

El sistema muestra un mensaje en la pantalla conforme no hay conexión a Internet.

4b – El usuario selecciona el botón de mapa:

El sistema abre la aplicación Google Maps y muestra solamente la ubicación de la persona necesitada, sin mostrar la ruta.

Requisititos especiales:

Para visualizar correctamente la lista de mapas hace falta una conexión a Internet.

Caso de uso 11: Visualizar perfil del donante en el Liquid Galaxy

Actores: usuario

Precondición: el usuario dispone de un sistema Liquid Galaxy conectado a la aplicación

Actor	Sistema	Backend	Liquid Galaxy
1. El usuario quiere visualizar un perfil de donante de una ciudad determinada			
	2. El sistema muestra una pantalla en la que el usuario puede escoger una ciudad		
		3. El backend devuelve las ciudades en las que hay registradas personas necesitadas	
4. El usuario escoge una ciudad			
			5. El Liquid Galaxy "vuela" a la ciudad seleccionada
	6. El sistema muestra una pantalla en la que el usuario puede escoger entre Personas necesitada, Donantes o Voluntarios		
7. El usuario selecciona Donantes			
		8. El backend devuelve todos los donantes registrados en la ciudad escogida anteriormente	
			9. El Liquid Galaxy muestra marcadores para todos los donantes existentes en la ciudad
	10. El sistema muestra en la pantalla una lista con todos los donantes existentes en la ciudad escogida, cada uno con dos opciones: Transacciones u Orbita		

11. El usuario selecciona un donante de la lista.			
		12. El backend devuelve la información del usuario seleccionado.	
			13. El Liquid Galaxy muestra un cuadro con la información básica del usuario seleccionado.

Tabla 20: Caso de uso 11: Visualizar perfil donante en el Liquid Galaxy

Cursos alternativos:

10a – El usuario selecciona el botón *Transacciones* para el donante:

El Liquid Galaxy muestra un cuadro adicional con información sobre las transacciones realizadas para este usuario.

10b – El usuario selecciona el botón *Orbita* para el donante:

El Liquid Galaxy gira entorno a las coordenadas donde se encuentra el donante.

5a, 9a, 13a – La conexión con el Liquid Galaxy no se ha establecido correctamente:

Los datos no se mostrarán hasta que la conexión es correcta.

Requisititos especiales:

Para visualizar un perfil de donante en el Liquid Galaxy hace falta una conexión a Internet, un sistema Liquid Galaxy y la conexión funcionando entre la aplicación y el sistema Liquid Galaxy.

Caso de uso 12: Visualizar perfil del voluntario en el Liquid Galaxy

Actores: usuario

Precondición: el usuario dispone de un sistema Liquid Galaxy conectado a la aplicación

Actor	Sistema	Backend	Liquid Galaxy
1. El usuario quiere visualizar un perfil de voluntario de una ciudad determinada			
	2. El sistema muestra una pantalla en la que el usuario puede escoger una ciudad		
		3. El backend devuelve las ciudades en las que hay registradas personas necesitadas	
4. El usuario escoge una ciudad			
			5. El Liquid Galaxy "vuela" a la ciudad seleccionada
	6. El sistema muestra una pantalla en la que el usuario puede escoger entre Personas necesitada, Donantes o Voluntarios		
7. El usuario selecciona Voluntarios			
		8. El backend devuelve todos los voluntarios registrados en la ciudad escogida anteriormente	
			9. El Liquid Galaxy muestra marcadores para todos los voluntarios existentes en la ciudad
	10. El sistema muestra en la pantalla una lista con todos los voluntarios existentes en la ciudad escogida, cada uno con dos		

	opciones: Transacciones u Orbita		
11. El usuario selecciona un voluntario de la lista.			
		12. El backend devuelve la información del usuario seleccionado.	
			13. El Liquid Galaxy muestra un cuadro con la información básica del usuario seleccionado.

Tabla 21: Caso de uso 12: Visualizar perfil voluntario en el Liquid Galaxy

Cursos alternativos:

10a – El usuario selecciona el botón *Transacciones* para el voluntario:

El Liquid Galaxy muestra un cuadro adicional con información sobre las transacciones realizadas para este usuario.

10b – El usuario selecciona el botón *Orbita* para el voluntario:

El Liquid Galaxy gira entorno a las coordenadas donde se encuentra la persona seleccionada.

5a, 9a, 13a – La conexión con el Liquid Galaxy no se ha establecido correctamente:

Los datos no se mostrarán hasta que la conexión es correcta.

Requisititos especiales:

Para visualizar un perfil de voluntario en el Liquid Galaxy hace falta una conexión a Internet, un sistema Liquid Galaxy y la conexión funcionando entre la aplicación y el sistema Liquid Galaxy.

Caso de uso 13: Visualizar Demo

Actores: usuario

Precondición: el usuario dispone de un sistema Liquid Galaxy conectado a la aplicación

Actor	Backend	Liquid Galaxy
1. El usuario quiere visualizar la Demo en el Liquid Galaxy. Selecciona el botón Demo en la pantalla principal del sistema		
	2. El backend devuelve la información necesaria	
		3. El Liquid Galaxy muestra una demo de la aplicación, pasando por cada ciudad y mostrando los tipos de usuarios registrados.
4. El usuario ha visualizado la Demo		

Tabla 22: Caso de uso 13: Visualizar Demo

Cursos alternativos:

1a, 2a – No hay conexión a Internet:

El sistema muestra un mensaje de error conforme no hay conexión a Internet.

1b – La conexión con el Liquid Galaxy no se ha establecido correctamente:

Los datos no se mostrarán hasta que la conexión es correcta.

Requisititos especiales:

Para visualizar la Demo en el Liquid Galaxy hace falta una conexión a Internet, un sistema Liquid Galaxy y la conexión funcionando entre la aplicación y el sistema Liquid Galaxy.

Caso de uso 14: Visualizar Estadísticas Globales

Actores: usuario

Precondición: el usuario dispone de un sistema Liquid Galaxy conectado a la aplicación

Actor	Backend	Liquid Galaxy
1. El usuario quiere visualizar las Estadísticas en el Liquid Galaxy. Selecciona el botón Estadísticas Globales en la pantalla principal del sistema		
	2. El backend devuelve la información necesaria	
		3. El Liquid Galaxy muestra un cuadro con información sobre las estadísticas de la base de datos.
4. El usuario ha visualizado las Estadísticas		

Tabla 23: Caso de uso 14: Visualizar Estadísticas

Cursos alternativos:

1a, 2a – No hay conexión a Internet:

El sistema muestra un mensaje de error conforme no hay conexión a Internet.

1b – La conexión con el Liquid Galaxy no se ha establecido correctamente:

Los datos no se mostrarán hasta que la conexión es correcta.

Requisititos especiales:

Para visualizar la Demo en el Liquid Galaxy hace falta una conexión a Internet, un sistema Liquid Galaxy y la conexión funcionando entre la aplicación y el sistema Liquid Galaxy.

Caso de uso 15: Visualizar Estadísticas locales

Actores: usuario

Precondición: el usuario dispone de un sistema Liquid Galaxy conectado a la aplicación

Actor	Sistema	Backend	Liquid Galaxy
1. El usuario quiere visualizar las estadísticas locales en el Liquid Galaxy. Selecciona el botón Estadísticas Locales en la pantalla principal del sistema			
		2. El backend devuelve la información necesaria	
			3. El Liquid Galaxy muestra un tour por cada ciudad, mostrando las estadísticas locales de cada una.
4. El usuario ha visualizado las estadísticas locales.			

Tabla 24: Caso de uso 15: Visualizar Tour

Cursos alternativos:

1a, 2a – No hay conexión a Internet:

El sistema muestra un mensaje de error conforme no hay conexión a Internet.

1b – La conexión con el Liquid Galaxy no se ha establecido correctamente:

Los datos no se mostrarán hasta que la conexión es correcta.

Requisititos especiales:

Para visualizar la Demo en el Liquid Galaxy hace falta una conexión a Internet, un sistema Liquid Galaxy y la conexión funcionando entre la aplicación y el sistema Liquid Galaxy.

Anexo 2: Formularios utilizados en las pruebas de usabilidad

Formulario Pre-Evaluación

1. Rango de edad: ☐ entre 17 y 23 ☐ entre 23 y 35 ☐ >35

2. Con que frecuencia utilizas el dispositivo móvil?

☐ más de una hora al día

☐ menos de una hora al día

☐ algunos días a la semana

☐ ocasionalmente

3. Tienes experiencia usando herramientas de programación?

☐ No

☐ Sí, nivel bajo o medio

☐ Si, nivel alto

4. Alguna vez has utilizando una aplicación que se dedica a ayudar a personas necesitadas?

☐ Si

☐ No

Preguntas Post-Evaluación

1. Deja tu opinión y sugerencias en relación al diseño de la aplicación.

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

2. Deja tu opinión y sugerencias en relación con las funcionalidades de la aplicación.

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Anexo 3 Resultados pruebas de usabilidad

Tarea 1: Regístrate como donante

- ***Información recolectada:***

	Sesión 1	Sesión 2
Total usuarios que han realizado la tarea	16	16
Tareas completadas con éxito	16	16
Tareas fallidas	0	0
Tiempo estimado en realizar la tarea para todos los usuarios	740s	715s

Tabla 25: Información recolectada tarea 1

- ***Medidas recolectadas:***

	Sesión 1	Sesión 2
Efectividad de suceso	1	1
Efectividad de fallos	0	0
Tiempo promedio por tarea	46.25s	44.68s

Tabla 26: Medidas recolectadas tarea 1

Tarea 2: Realiza una donación personalmente a un usuario que consideras

- ***Información recolectada:***

	Sesión 1	Sesión 2
Total usuarios que han realizado la tarea	16	16
Tareas completadas con éxito	16	16
Tareas fallidas	0	0
Tiempo estimado en realizar la tarea para todos los usuarios	893s	850s

Tabla 27: Información recolectada tarea 2

- ***Medidas recolectadas:***

	Sesión 1	Sesión 2
Efectividad de suceso	1	1
Efectividad de fallos	0	1
Tiempo promedio por tarea	55.82s	53.12s

Tabla 28: Medidas recolectadas tarea 2

Tarea 3: Realiza una donación a través de un voluntario a un usuario que consideras

- ***Información recolectada:***

	Sesión 1	Sesión 2
Total usuarios que han realizado la tarea	16	16
Tareas completadas con éxito	16	16
Tareas fallidas	0	0
Tiempo estimado en realizar la tarea para todos los usuarios	1200s	1050s

Tabla 29: Información recolectada tarea 3

- ***Medidas recolectadas:***

	Sesión 1	Sesión 2
Efectividad de suceso	1	1
Efectividad de fallos	0	1
Tiempo promedio por tarea	75s	66.625s

Tabla 30: Medidas recolectadas tarea 3

Tarea 4: Consulta el mapa de las personas necesitadas

- ***Información recolectada:***

	Sesión 1	Sesión 2
Total usuarios que han realizado la tarea	16	16
Tareas completadas con éxito	16	16
Tareas fallidas	0	0
Tiempo estimado en realizar la tarea para todos los usuarios	243.2s	200s

Tabla 31: Información recolectada tarea 4

- ***Medidas recolectadas:***

	Sesión 1	Sesión 2
Efectividad de suceso	1	1
Efectividad de fallos	0	1
Tiempo promedio por tarea	15s	12.5s

Tabla 32: Medidas recolectadas tarea 4

Tarea 5: Consulta la lista de mapas de las personas necesitadas

- ***Información recolectada:***

	Sesión 1	Sesión 2
Total usuarios que han realizado la tarea	16	16
Tareas completadas con éxito	16	16
Tareas fallidas	0	0
Tiempo estimado en realizar la tarea para todos los usuarios	179.2s	180s

Tabla 33: Información recolectada tarea 5

- ***Medidas recolectadas:***

	Sesión 1	Sesión 2
Efectividad de suceso	1	1
Efectividad de fallos	0	1
Tiempo promedio por tarea	11.2s	11.25s

Tabla 34: Medidas recolectadas tarea 5

Tarea 6: Realiza una donación para los desarrolladores

- ***Información recolectada:***

	Sesión 1	Sesión 2
Total usuarios que han realizado la tarea	16	16
Tareas completadas con éxito	10	12
Tareas fallidas	6	4
Tiempo estimado en realizar la tarea para todos los usuarios	723.2s	685s

Tabla 35: Información recolectada tarea 6

- ***Medidas recolectadas:***

	Sesión 1	Sesión 2
Efectividad de suceso	0.625	0.75
Efectividad de fallos	0.375	0.25
Tiempo promedio por tarea	45.2 s	42.81s

Tabla 36: Medidas recolectadas tarea 6

Tarea 7: Regístrate como voluntario

- ***Información recolectada:***

	Sesión 1	Sesión 2
Total usuarios que han realizado la tarea	16	16
Tareas completadas con éxito	16	16
Tareas fallidas	0	0
Tiempo estimado en realizar la tarea para todos los usuarios	723.2s	620s

Tabla 37: Información recolectada tarea 7

- ***Medidas recolectadas:***

	Sesión 1	Sesión 2
Efectividad de suceso	1	1
Efectividad de fallos	0	0
Tiempo promedio por tarea	45.2 s	38.75s

Tabla 38: Medidas recolectadas tarea 7

Tarea 8: Crea un perfil nuevo para una persona necesitada

- ***Información recolectada:***

	Sesión 1	Sesión 2
Total usuarios que han realizado la tarea	16	16
Tareas completadas con éxito	12	14
Tareas fallidas	4	2
Tiempo estimado en realizar la tarea para todos los usuarios	2004.8s	1680s

Tabla 39: Información recolectada tarea 8

- ***Medidas recolectadas:***

	Sesión 1	Sesión 2
Efectividad de suceso	0.75	0.875
Efectividad de fallos	0.25	0.125
Tiempo promedio por tarea	125.3s	105s

Tabla 40: Medidas recolectadas tarea 8

Tarea 9: Edita la información para este perfil y cambia la historia de vida

- ***Información recolectada:***

	Sesión 1	Sesión 2
Total usuarios que han realizado la tarea	16	16
Tareas completadas con éxito	10	14
Tareas fallidas	6	2
Tiempo estimado en realizar la tarea para todos los usuarios	720s	690s

Tabla 41: Información recolectada tarea 9

- ***Medidas recolectadas:***

	Sesión 1	Sesión 2
Efectividad de suceso	0.625	0.875
Efectividad de fallos	0.375	0.125
Tiempo promedio por tarea	45s	43.12s

Tabla 42: Medidas recolectadas tarea 9

Tarea 10: Contacta un donante para realizar la entrega de una donación

- ***Información recolectada:***

	Sesión 1	Sesión 2
Total usuarios que han realizado la tarea	16	16
Tareas completadas con éxito	11	15
Tareas fallidas	5	1
Tiempo estimado en realizar la tarea para todos los usuarios	896s	890s

Tabla 43: Información recolectada tarea 10

- ***Medidas recolectadas:***

	Sesión 1	Sesión 2
Efectividad de suceso	0.687	0.937
Efectividad de fallos	0.312	0.062
Tiempo promedio por tarea	56s	55.62s

Tabla 44: Medidas recolectadas tarea 10

Tarea 11: Contacta con nosotros

- ***Información recolectada:***

	Sesión 1	Sesión 2
Total usuarios que han realizado la tarea	16	16
Tareas completadas con éxito	16	16
Tareas fallidas	0	0
Tiempo estimado en realizar la tarea para todos los usuarios	720s	650s

Tabla 45: Información recolectada tarea 11

- ***Medidas recolectadas:***

	Sesión 1	Sesión 2
Efectividad de suceso	1	1
Efectividad de fallos	0	0
Tiempo promedio por tarea	45s	40.62s

Tabla 46: Medidas recolectadas tarea 11

Tarea 12: Entra en la parte del Liquid Galaxy y mira la información relacionada con una persona necesitada que escoges

- ***Información recolectada:***

	Sesión 1	Sesión 2
Total usuarios que han realizado la tarea	7	7
Tareas completadas con éxito	5	6
Tareas fallidas	2	1
Tiempo estimado en realizar la tarea para todos los usuarios	455s	500s

Tabla 47: Información recolectada tarea 12

- ***Medidas recolectadas:***

	Sesión 1	Sesión 2
Efectividad de suceso	0.714	0.857
Efectividad de fallos	0.285	0.142
Tiempo promedio por tarea	65s	71.42s

Tabla 48: Medidas recolectadas tarea 12